The
SENSIBLE
SOLUTION™

Language

O'HANLON
COMPUTER SYSTEMS

COPYRIGHT © 1984 by
O'Hanlon Computer Systems, Inc.
All rights reserved.

O'Hanlon Computer Systems, Inc.
Opportunity Building
8383 158th Avenue N.E.
Redmond, Washington 98052

July 1984

Copyright (c) 1984 by O'Hanlon Computer Systems, Inc.
All rights reserved.

No part of this publication may be reproduced in any form without prior written permission from O'Hanlon Computer Systems, Inc. The SENSIBLE SOLUTION, The SENSIBLE SOLUTION Language, and The SENSIBLE SOLUTION Management Series are all registered trademarks of O'Hanlon Computer Systems,Inc.

CP/M and MP/M are registered trademarks of Digital Research Corp. MSDOS is a registered trademark of Microsoft, Inc. DPC/OS is a registered trademark of Action Computer Enterprises. MmmOST is a registered trademark of TeleVideo. TurboDOS is a registered trademark of Software 2000. n/STAR is a registered trademark of Molecular Computer.

Mail Address:

Customer Support Services
O'Hanlon Computer Systems
Opportunity Building
8383 158th Avenue N.E.
Redmond, Washington 98052

8. Prior to calling your dealer or O'Hanlon Computer Systems, please review the following check list, review your problems, and locate the appropriate sections in the reference manual.

Before You Call Customer Support

Prior to calling your dealer or O'Hanlon Computer Systems for support, there are several things you can do that will result in quicker and better support service:

1. You may be asked for the following information:

   A. Your Hardware and Operating System.

   B. If you were in the SENSIBLE SOLUTION **Language** or if you were executing a SENSIBLE SOLUTION **Application.**

   C. The specific product/application **Version** and/or serial number.

2. You must be familiar with your hardware and operating system prior to contacting Customer Support. For example, you may be asked to transfer files, load disks, re-boot the system, etc.

3. You must be familiar with the O'Hanlon Computer System Software Manual(s). It is often very helpful if you can identify which sections of the manual relate to your problems-note the page numbers.

4. If you are a new user and Customer Support suspects that the installation is not correct, you may be asked to re-install the software. Please read your installation manual carefully and have it available for reference

5. If you are operating in a multi-user environment, you should

know your operating system status and the status of all other users on your system.

6. If an error message is displayed or it returns abruptly to the operating system:

   A. Note the exact error message; character by character, along with any drive letter designation or file name.

   B. Note the immediate sequence of events and conditions just prior to the error. Note the menu selections, the cursor location on your screen, and the data last entered.

7. We recommend that you have your telephone, terminal, and computer close by so that you can try alternate procedures that we may suggest over the phone.

The SENSIBLE SOLUTION

System Requirements

The following are the basic system requirements for the installation and succcessful operation of **SENSIBLE SOLUTION** software:

(1) **Operating System:** SENSIBLE SOLUTION requires a **CP/M, MP/M, MS-DOS** or similar, compatible operating system. Most hardware which supports such an operating system will run SENSIBLE SOLUTION. Examples of these operating systems are: CP/M, MP/M, MS-DOS, DPC/OS, TurboDOS, PC-DOS, MmmOST, N-Star, CP-NET.

(2) **RAM Memory:** SENSIBLE SOLUTION requires RAM memory of 48k TPA or greater (free user area exclusive of operating system requirements):

(a) CP/M, MP/M	48k RAM TPA
(b) CP/M (DEC Rainbow/Pro)	96k
(c) PC-DOS, MS-DOS (IBM, TI)	128k
(d) MS-DOS (Victor)	256k

(3) **Mass Storage:** SENSIBLE SOLUTION requires mass storage capability of at least two floppy Disk Drives, each with at least **300k bytes** usable (floppy disk) storage capacity (after formating). Additional drives, disk capacity, and/or hard disks will increase system performance. Hard disks are recommended for multi-user implementation and/or extensive applications.

(4) **CRT Terminal:** SENSIBLE SOLUTION requires a CRT (Video) terminal of the following minimum requirements:

(a) ASCII serial type or ANSI compatible
(b) Screen Display: 24 (lines) by 80 (columns)
(c) Direct Cursor Addressing (absolute)
(d) Clear to End of Line
(e) Clear Screen

(5) **Printer:** SENSIBLE SOLUTION requires a printer with the following minimum requirements:
(a) ASCII type
(b) 80 column or more (e.g. 255 column compressed print)

O'Hanlon Computer Systems, Inc.

Customer Support Policy

1. Customer support services are available only for O'Hanlon Computer Systems software products. (Note: Although all of our support personnel are very knowledgable, they may not be experts on your specific operating system or your hardware!)

2. The initial support responsibility for O'Hanlon Computer Systems Software Products lies with the dealer through which you purchased your software. Please contact your dealer first for resolution of problems and questions.

3. O'Hanlon Computer Systems customer support is limited to SENSIBLE SOLUTION software. User modifications to O'Hanlon applications and user application programs are NOT supported. If help is required in this area, contact O'Hanlon Computer Systems for Custom Software Consulting.

4. O'Hanlon Computer Systems software is supported for a period of 90 days after shipment.

5. Up to two hours of support time (telephone and research) is available without cost during the first 90 days after shipment. As long as the software maintenance contract is maintained, up to two hours of support is available per quarter (non-cumulative). After that, Customer Support time is billable at the current O'Hanlon Computer Systems rate.

6. Only the current version of the software is supported. Those users with older software are required to obtain a current version.

7. The O'Hanlon Computer Systems Customer Support telephone number is:

206-885-2502 Please ask for Customer Support.

Hours are Morning: 8:00 to 12:00
Afternoon: 1:00 to 4:00 Pacific Time
Monday thru Friday

(No collect calls accepted)

The logical screen being displayed on your monitor.

Software
The instructions (programs) that direct the operation of the computer hardware.

Source code
The program you write with the SENSIBLE SOLUTION Language. The source code is then compiled by the computer and turned into pseudo code, a set of instructions that the computer can more readily understand.

String
A contiguous series of alphanumeric characters. 'Ac%!L+' is a string.

Syntax
The rules that decide how programming language statements must be constructed. In other words, the grammar of the language.

Toggle
A two position switch that changes from one position to the other every time it is operated. In computer language it can be a 'logical' item (could be a bit) that changes its status back and forth every time some condition is satisfied.

Value
A number or a name (string of characters) given to a field. For example, the field 'Name' can have the value 'Smith' or 'Brown'; the field 'amount' can have the value '234.56'.

Variable
A unit that can take different values at different times. In SENSIBLE SOLUTION the word variable has the same meaning as field.

The SENSIBLE SOLUTION

System Specifications

Maximum Program Size	O/S Limited
Maximum Data File Size	O/S Limited
Maximum Number of Data Files	Unlimited
Maximum Number of Records per Data File	16,777,216
Maximum Number of Data Fields per Record	1,000
Maximum Bytes per Data File Record	Memory Limited
Maximum Number of Open Files in a Program	16
Maximum Number of Indexes (Keys) per Data File Record	10
[This includes One Pre-Defined Record Number Index]	
Maximum Number of Keys per Screen or Program	100
Maximum Length of Key Field	72
Maximum Length of a Single Field	255
Stored Number Range:	
Maximum	+99,999,999,999.9999
Minimum	- 9,999,999,999.9999
Decimal Place Precision	4
[Computations are done to 5 decimal place precision, then rounded to the precision of the target field.]	
Maximum Number of Accessed fields per Program	255
Maximum Number of Command Lines per Program	2,000
Maximum Number of Command Line Labels per Program	300
Maximum Number of Nested Subroutines (GOSUB)	20
Maximum Length of Reporter Print Line	Printer Limited
Maximum Number of Report Format Lines	60
Maximum Fields (fields) on a Screen/Report Format	255
Maximum Length of Field (Variable) Name	15
Maximum Number of Unique Field Names per Program	255

NOTES:

O/S Limited: Limited by the disk capacity and operating system.

File name
The first part of the file specifier. SENSIBLE is the name of the file SENSIBLE.COM.

File specifier
The characters used to fully identify a file, the file name and the file extension separated by a period. SENSIBLE.COM is the file specifier of the file whose name is SENSIBLE and whose extension is COM.

Format
The logical organization of data within the computer memory or on disk.

Hardware
The mechanical, electronic and electrical devices that make up the computer.

Key, control
See control key.

Key, index
A logical attribute given to one or more fields in a file to permit fast access and retrieval of a record given the value of the key.

Key, of keyboard
What you press to generate a character on the screen of your computer.

Label
A descriptive identifier. Typically, a label is a group of characters used to identify a parameter on the screen. In a program, labels identify the line to which program control is to pass when certain conditions are satisfied.

Listing, program
A list of the instructions contained in the program. It can be on paper (printer listing), or on the screen (CRT listing).

Logon or logged onto
The act of assigning a physical disk drive to a logical location in the computer's operating system. Usually disk drives are represented by alphabetic values from A to P. To gain access to a physical disk drive (to logon to a disk drive) you would normally enter the disk drive location followed by a colon. For example, **typing C:** and the carriage return key will log you onto drive C.

Machine code
A series of instructions in a form the computer can understand directly without any translation.

Memory
The place where the computer stores information for immediate and fast access. It is often called RAM -- Random Access Memory.

Monitor
The physical screen of your computer.

Numeric character
Number characters only.

Operating system
The software that allows your computer to communicate with your CRT, disk drives, printer, etc.. In our case, the operating system allows SENSIBLE SOLUTION to communicate with these physical devices.

Parameter
An item of information that can be changed in accordance with your wishes. For instance, you might have a parameter called 'Payment amount', which would give different values during execution of your program.

Program
An ordered list of instructions directing a computer to carry out a desired sequence of operations.

Pseudo code
The code, meaningless to humans, which the computer generates from a program written in SENSIBLE SOLUTION. Pseudo code is not machine code (which the computer can understand immediately), but is fairly close to it.

Record
A series of related data that is created by a program and can be interpreted by a program. Many records form a file.

Record, physical
The space allocated on disk or in memory to store records. Its length is fixed.

Screen
The physical device that is used by the computer to display information. It is often referred to as monitor or CRT. A screen can also be interpreted to mean the template of information that defines the position of labels, field windows and comments as they will be displayed on the CRT when a program is run.

Screen, currently active

Byte
A group of adjacent bits, nowadays generally accepted as 8 bits.

Compile
The coding operations the computer performs to convert your program lines (something that you can read and understand) into something the computer can understand.

Constant or literal
A parameter whose value is fixed throughtout the execution of a program.

Control character
A special one-character code that is generally not displayed on the screen that can interpreted by a program to initiate some action. It is generated by holding down the control key while another character key on the keyboard is depressed.

Control code
A special code which sets certain functions within the computer. A certain control code can blank your computer screen for instance. It can be generated from the keyboard or from the software.

Control key
A special keyboard key to be used in conjunction with an ordinary character key to send special commands to the computer. On some keyboards it is marked CTRL or ALT. Control keys are used for screen handling functions. When the cotrol key is depressed in conjunction with another key, it generates a control character.

CRT
The physical screen of your computer. CRT stands for Cathode Ray Tube.

Data file
A file of data. It contains the 'data', the information that you have put into the computer that you want to manipulate or simply store and retrieve. A list of names and addresses for instance.

Debug
To isolate and correct errors in a computer program.

Default drive
The disk drive that is automatically accessed by the operating system when no other drive is specified; normally indicated on your screen with a letter from A to P followed by an angle bracket or a dot. The disk drive you have logged onto and are running programs from.

Default value
The value that is automatically assigned to a field by a program.

Disk
Plastic or metal disk, coated with magnetic material on which data can be stored and retrieved by the computer. The disk is divided into concentric rings called tracks, each of which is in turn subdivided into sectors. The common varieties are floppies and hard. Floppies are flexible plastic disks contained in an envelope. Their data storage capacity is fairly limited and their life is affected by the continuous friction with the disk drive read/write head and atmospheric dust. Read and write operations are slow in comparison with hard disks. Hard disks have a much larger capacity than floppies, typically from 10 to 30 times, and they operate at a far greater speed. They are not affected by dust and dirt as they are contained in sealed enclosures.

Disk drive
The machine that spins the disk and reads its contents. A disk drive is normally identifed with a letter of the alphabet from A to P followed by a colon, i.e. A:

Executive program
A program that has been compiled and is ready to be run. It will perform all the instructions given by the programmer.

Field
A subdivision of a record. A field is an area where data of a certain type is stored or found as a single entity. In SENSIBLE SOLUTION field is equivalent to variable.

Field, window
The area on the screen, to the right of a field label designated to display the value of the field.

File
An organized and structured collection of information. The information is composed of records and each record is composed of fields.

File, data
See data file.

File extension
The second part of the file specifier. COM is the extension of file SENSIBLE.COM.

data entry and update screen for the user, etc..

The SENSIBLE SOLUTION is a procedure oriented language that can easily perform these kinds of tasks. With the SENSIBLE SOLUTION, the relatively inexperienced computer user can perform the arcane art of business applications programming.

Begin your study of SENSIBLE SOLUTION by reading over this manual carefully. At this stage, pay particular attention to this introductory section and then read the INSTALLATION section at the back of this manual. Before you run SENSIBLE SOLUTION on your computer system, you will need to perform a short installation procedure described in the INSTALLATION section of this manual.

By signing and returning the User License Agreement you will be entitled to the SENSIBLE SOLUTION warranty and return rights.

Before you go one step further, make back-up copies of your new SENSIBLE SOLUTION diskettes. If you don't know how to do this, there should be detailed instructions on disk copying in your computer hardware system manual.

GLOSSARY

There aren't too many businesses that have generated as much jargon as the computer industry. The rapid changes within this industry lead to a constantly changing vocabulary. Confusion results not only because the new terms may not be easy for all participants to understand, but also because the same term may be used in more than one way.

This glossary is aimed at the non-specialist computer user and we hope that it will help to dispel such confusion by bringing together all the jargon words used throughout this manual. All words underlines are referred to elsewhere in this glossary.

Alphanumeric, character
Any character that can be a number, a letter or other symbols like punctuation marks and mathematical signs.

Application, package
A series of programs that satisfy a given requirement. A hotel booking system is a possible application of SENSIBLE SOLUTION for instance.

Array
A set of fields identified by a common name. In SENSIBLE SOLUTION arrays have two dimensions: the length of the field and the number of fields. All fields must have the same length.

Bit
The commonly used abbreviation of 'binary digit'. It represents the smallest unit of information that can be stored in a computer, either a 0 or a 1.

Boot
The operation of transferring the operating system from the disk to the computer memory.

Buffer
Memory storage area where information is assembled before transmission to permanent storage.

The SENSIBLE SOLUTION Language

The powerful, inexpensive microcomputer has brought a revolution to the business office, but all too often it seems the new computer wins the battle and loses the war! A computer without a program is just an overpriced paperweight. A computer with a **poor** program is **worse** than useless: you fight to get the data into the system, work around the parts the computer doesn't know how to do, and maybe you finally give up in frustration and go back to doing the job by hand.

Computers have no imagination. They must be exactingly instructed, down to the most nit-picking detail, how to accomplish a task. A few years ago, one misplaced comma in a program sent the Voyager III space probe careening into the sun.

This is why designing business systems with languages like BASIC or COBOL is a tedious, time-consuming and expensive affair. Even a simple General Ledger can take months to design, code, test and debug; adding one tiny feature to the finished system can send the whole thing back to the drawing board.

Of course, there are hundreds of "standard" business packages already available. The problem is, nobody runs a standard business. There's always some billing system the boss wants to use, a special arrangement with an old customer or something the sales staff is used to doing and won't change. Often the accountant keeps track of the exceptions by hand and the two sets of books slowly drift away from any relation to each other. The computer becomes more trouble than it's worth.

Database management programs are a recent rage because they let you design your own systems. It's a good start, but many of these programs have limited accounting capability and poor facilities for validation. Designing systems with a database program may be as difficult as using BASIC. Worst of all, each new system is totally unrelated to earlier ones -- you can't build on what you've already done. What you **have** is a lot of pieces that do part of the job but can't work together. What you **need** is a system that puts them all together:

- a quick and easy way to create screens and reports as they will appear on a terminal or printed page.

- a data control system that keeps track of all the items of information: what they are, what they contain, what is valid data and how each item relates to all the others.

- a set of standardized tools so the designer doesn't have to rebuild the same logic over and over yet allows the operator to use the same data-entry techniques for each and every application.
- a program generator that handles the nit-picking details while leaving the designer totally free to specify the logic of the system.
- a structure that lets the designer change or expand the system without having to retest everything again from the ground up.
- finally, a resulting system that is fast, efficient, and accurately handles the needs of a full-size business.

You've probably realized that this isn't just a wish list. O'Hanlon Computer Systems has developed exactly this system: a program that generates the full range of business applications from mailing lists to accounting packages to forecasting to cost control or whatever you can imagine. We have developed a system that puts the power of the microcomputer to work for you **today.** That's why we call it The SENSIBLE SOLUTION.

Most computer languages like Basic, Fortran, and Pascal are general purpose syntax oriented languages. All of these 'high level' languages provide many diverse yet primitive functions that allow you to write programs for scientific, engineering, and business applications.

There is a world of difference, however, between a scientific 'number crunching' program where you might encounter numbers running from millions to millionths and a business application program that keeps track of numbers to two decimal places. The point is, high level languages provide the tools for an experienced programmer to write many different kinds of applications, but this diversity is usually paid for at the expense of speed and ease of use.

Since business programs are usually limited in scope, using the same kinds of procedures over and over, why use a diverse, high level language to solve a specific business problem? There is a better way, a procedural language composed of the kinds of procedures that you use over and over -- create a file system, send and retrieve data from a file, update a file, provide a

INSTALLATION MANUAL

The SENSIBLE SOLUTION Language
Table of Contents

INTRODUCTION

TUTORIAL

REFERENCE MANUAL

MENU SELECTIONS

LANGUAGE COMMANDS

DATA STRUCTURES

APPENDIX

PROGRAM IS FURNISHED, TO BE FREE FROM DEFECTS IN MATERIALS AND WORKMANSHIP UNDER NORMAL USE FOR A PERIOD OF THIRTY (30) DAYS FROM THE DATE OF INITIAL DELIVERY TO INITIAL LICENSEE AS EVIDENCED BY A COPY OF RECEIPT THEREFORE.

4. LIMITATIONS OF REMEDIES: O'HANLON'S ENTIRE LIABILITY AND YOUR EXCLUSIVE REMEDY SHALL BE:

A. THE REPLACEMENT OF ANY DISKETTE OR CASSETTE NOT MEETING THE O'HANLON "LIMITED WARRANTY", WHICH IS RETURNED TO O'HANLON OR AN AUTHORIZED O'HANLON DEALER WITH A COPY OF YOUR USER LICENSE AGREEMENT AND RECEIPT THEREOF, OR

B. IF O'HANLON OR THE DEALER IS UNABLE TO DELIVER A REPLACEMENT DISKETTE OR CASSETTE WHICH IS FREE OF DEFECTS IN MATERIALS OR WORKMANSHIP, YOU MAY TERMINATE THIS AGREEMENT BY RETURNING THE PROGRAM AND YOUR MONEY WILL BE REFUNDED.

IN NO EVENT WILL O'HANLON BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS, OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE SUCH PROGRAM, EVEN IF O'HANLON OR AN AUTHORIZED O'HANLON DEALER HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY WHO MAY ASSERT OR CLAIM DAMAGES.

5. GENERAL: You may not sublicense, assign or transfer the license or the programs except as expressly provided in this license agreement. Any attempt otherwise to sublicense, assign or transfer any of the rights, duties or obligations hereunder is void.

O'Hanlon Computer Systems, Inc. agrees to supply you with product enhancements as developed. You agree to pay O'Hanlon Computer Systems, Inc. for the cost of the media upon which the enhancements are distributed, shipping & freight costs and a reasonable handling charge.

O'HANLON COMPUTER SYSTEMS, INC.
OPPORTUNITY BUILDING
8383 158th AVENUE N.E.
REDMOND, WASHINGTON 98052
(206) 885-2502

Update Procedure

SENSIBLE SOLUTION(tm) software is constantly undergoing enhancements and revisions. This is normal software maintenance. The current version of software that we produce may be a major or minor release in terms of its impact on the user. O'Hanlon Computer Systems will only give customer support to those users who are running the current major software release. The version number of your software will allow you to determine whether or not the current release is major. For example, Version 2.0C is a major release compared to Version 1.24. Thus, if O'Hanlon Computer Systems is currently shipping Version 2.0C and you are using Version 1.24, you must update your software in order to be eligible for software support. SENSIBLE SOLUTION(tm) dealers and end-users will be made aware of major releases through newsletters and bulletins.

Software maintenance contract fees are billed directly to the end-user by O'Hanlon Computer Systems. The fee is 15% of the suggested retail price billed annually. Any end-user who does not keep their software maintenance contract current will not be supported by O'Hanlon Computer Systems after the 90 day warranty period. End-users in this category can only receive a new version of SENSIBLE SOLUTION(tm) software by purchasing a new O'Hanlon Computer Systems Software Maintenance Contract at 30% of the current suggested retail price.

Software updates are handled by your dealer. Details are as follows:

1. A list of the end-users who have paid the Software Maintenance Contract are supplied to the SENSIBLE SOLUTION(tm) software dealers by O'Hanlon Computer Systems.

2. SENSIBLE SOLUTION(tm) dealers will usually require a nominal fee for end-user updates to defray shipping expenses and installation costs.

3. Your SENSIBLE SOLUTION(tm) dealer may elect to ship software updates COD. COD charges may include:
   (a) Media (cost of the diskettes, manuals, etc)
   (b) Shipping (freight) costs
   (c) Handling charge

O'Hanlon Computer Systems, Inc.

USER LICENSE AGREEMENT

You should carefully read the following terms and conditions before opening and accepting this diskette package.

1. O'Hanlon Computer Systems, Inc. provides the programs and operator manuals, and licenses their use in the generation and execution of software applications. Included in these programs are copyrighted materials of O'Hanlon, which are licensed by O'Hanlon to you for use under this license agreement. You assume responsibility for the selection of the program to achieve your intended results, and for the installation, use and results obtained from the program.

2. SOFTWARE MAINTENANCE CONTRACT If you so elect, you agree to pay O'Hanlon Computer Systems, Inc. an annual maintenance fee. The fee is currently billed annually, and commences 90 days after the date of purchase. This fee may be adjusted as deemed appropriate by O'Hanlon Computer Systems, Inc., but in no event shall the fee exceed 15% of the suggested retail price.

   If the Software Maintenance Contract fee is not paid when due, O'Hanlon Computer Systems, Inc. is under no obligation to provide continued support, maintenance or product enhancements. If a licensee desires reinstatement after non-payment, reinstatement will be made upon reasonable terms and conditions agreed between licensee (or transferee) and O'Hanlon Computer Systems, Inc., taking into account the period of non-payment, status of enhancements, and the then conditions of the product.

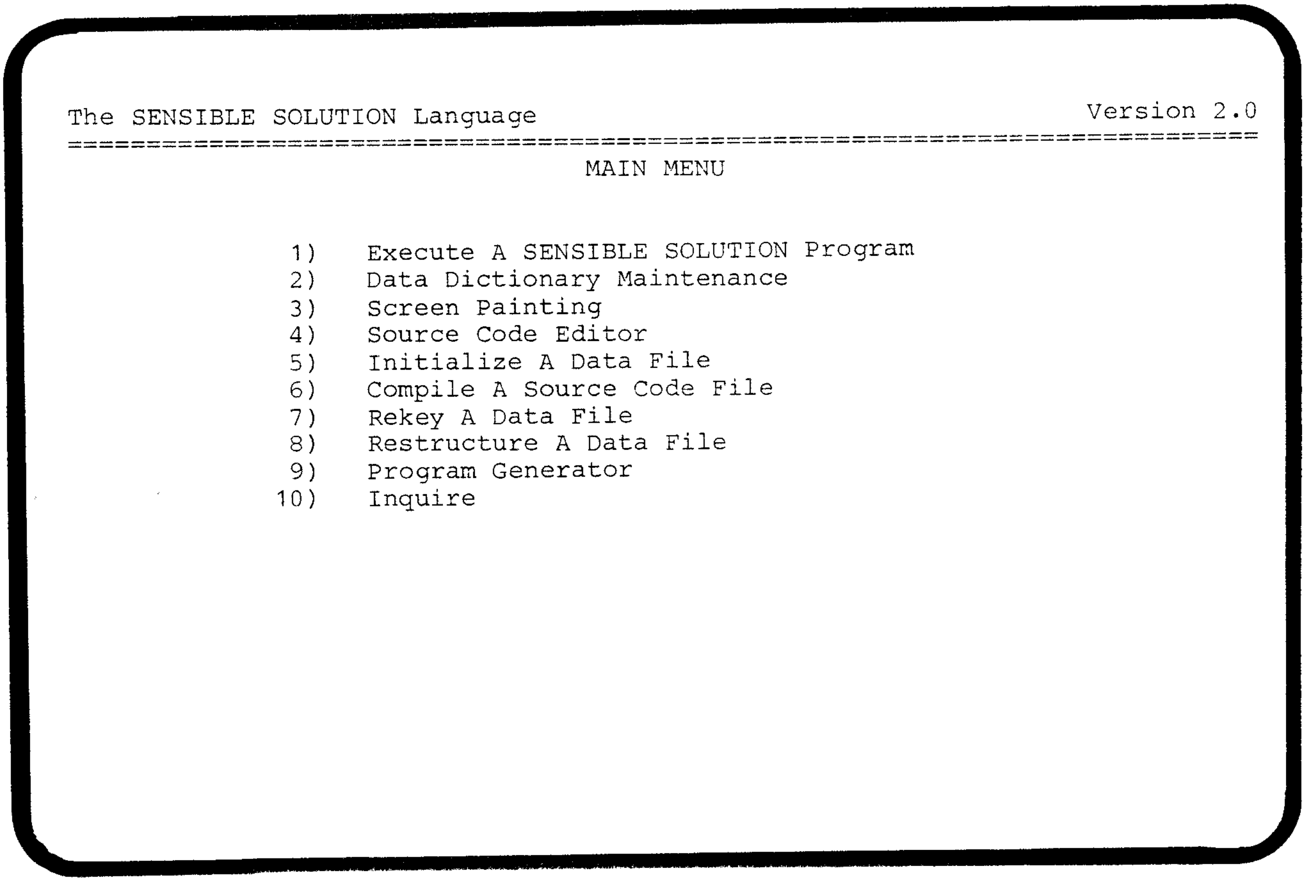
3. **LIMITED WARRANTY: THE PROGRAM IS PROVIDED WITH A LIMITED WARRANTY AS DESCRIBED BELOW. O'HANLON MAKES NO EXPRESS OR IMPLIED WARRANTY INCLUDING, BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE UNLESS SPECIFICALLY STATED. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU, SHOULD THE PROGRAM PROVE DEFECTIVE. YOU (AND NOT O'HANLON OR ANY AUTHORIZED O'HANLON DEALER), ASSUME THE ENTIRE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.**

   O'HANLON DOES NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE PROGRAM WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE PROGRAM WILL BE UNINTERRUPTED OR ERROR FREE.

   HOWEVER, O'HANLON WARRANTS THE DISKETTE(S) OR CASSETTES ON WHICH THE

The SENSIBLE SOLUTION Language

Main Menu Screen Display



```
The SENSIBLE SOLUTION Language                                    Version 2.0
==============================================================================
                                  MAIN MENU


                 1)   Execute A SENSIBLE SOLUTION Program
                 2)   Data Dictionary Maintenance
                 3)   Screen Painting
                 4)   Source Code Editor
                 5)   Initialize A Data File
                 6)   Compile A Source Code File
                 7)   Rekey A Data File
                 8)   Restructure A Data File
                 9)   Program Generator
                10)   Inquire
```

Execute A SENSIBLE SOLUTION Program

Overview:

This program is the primary SENSIBLE SOLUTION executive program. SENSIBLE.COM is the initial program to call from the operating system level to execute the SENSIBLE SOLUTION Language main menu and it is also the run-time executive program used to execute all SENSIBLE SOLUTION programs (filename.RUN).

Operation:

SENSIBLE.COM can be executed from the operating system level by typing the word SENSIBLE [RETURN]. When this is done, the files, SENSCTRL.MS and ERRFLE.MS/.KS, will immediately be accessed. SENSIBLE SOLUTION will read the SENSCTRL.MS file to determine the printer device and terminal device communication control codes that you previously specified with the system installation program -- SENSETUP.COM. SENSIBLE will then open ERRFLE.MS and execute the compiled SENSIBLE SOLUTION program, MENU.RUN. This program will immediately display the SENSIBLE SOLUTION Language menu shown on the next page.

Selection number 1 on the language menu, "Execute A SENSIBLE SOLUTION Program," will also call SENSIBLE.COM. Thus SENSIBLE.COM may be called from either the operating system level or from the language menu level.

Introduction

Welcome! We'd like to introduce you to The SENSIBLE SOLUTION, a language that lets you quickly and interactively design and modify business and database applications.

We want to demonstrate the ease with which you can design new applications, enhance old ones, and integrate applications into your particular business environment. Your task will be to create a collection of programs which can form the backbone of a customer order/entry/payment control system and could be integrated into a complete accounting system.

You will become familiar with the logical progression of steps taken to create a program. Using the selections on the Main Menu, you will:

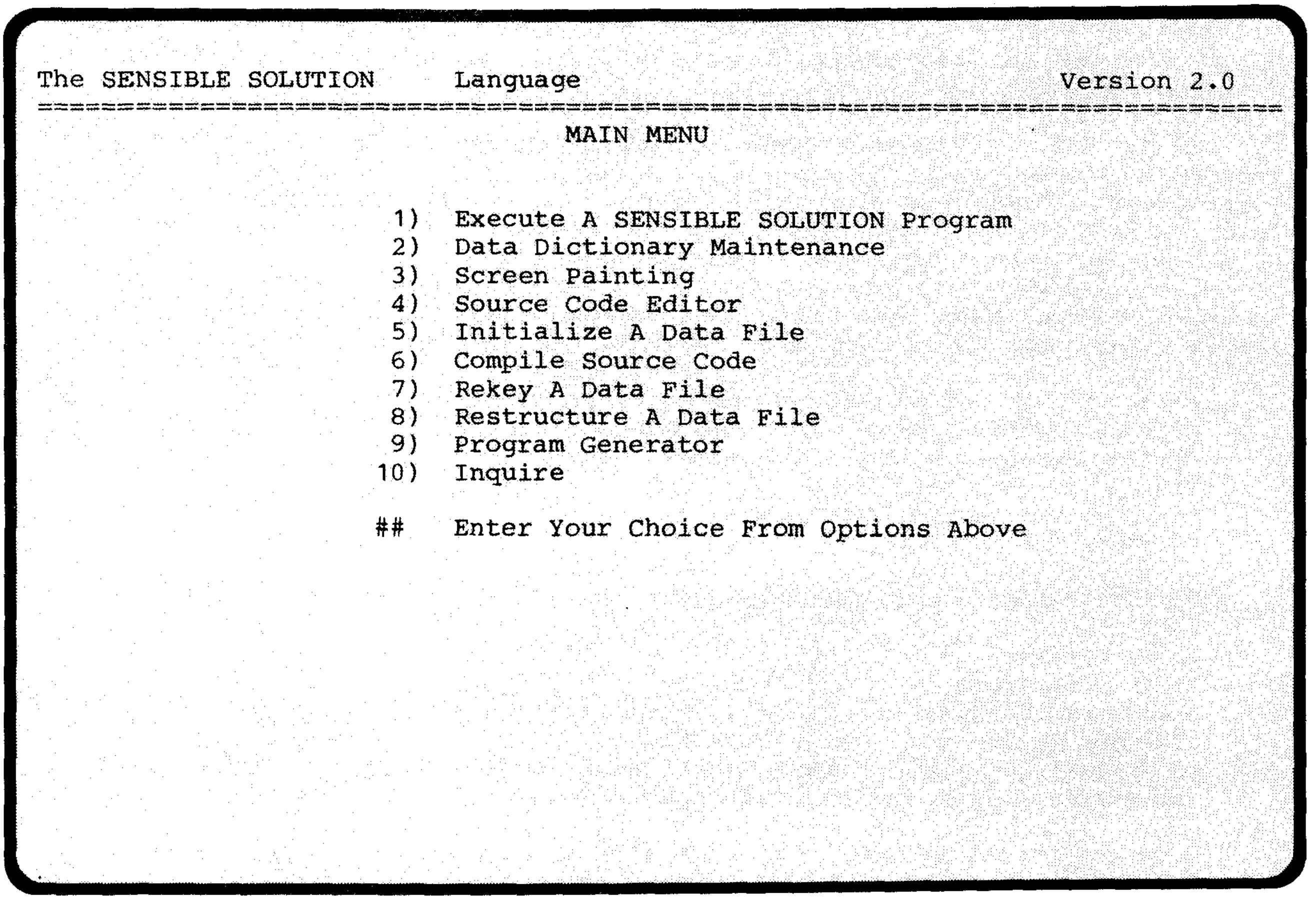
- create files and fields in the Data Dictionary
- paint program screens using the Screen Painter
- create the source code file using the Source Code Editor
- initialize the data files
- compile the source code file
- execute the program

In addition, we will will teach you about:

- using arrays
- data structures
- how easy it is to modify an existing program
- Inquire -- the data query facility
- creating report formats that will print reports which include information from several different files

If you're a novice, relax! We'll provide you with all the basics you need to familiarize yourself with SENSIBLE SOLUTION procedures and embark on your programming endeavor. And you pros, here's a quick overview of the entire SENSIBLE SOLUTION development system which will assist your enterprise.

After typing SENSIBLE **[RETURN]** --



```
The SENSIBLE SOLUTION     Language                                  Version 2.0
==============================================================================
                                 MAIN MENU


                           1)   Execute A SENSIBLE SOLUTION Program
                           2)   Data Dictionary Maintenance
                           3)   Screen Painting
                           4)   Source Code Editor
                           5)   Initialize A Data File
                           6)   Compile Source Code
                           7)   Rekey A Data File
                           8)   Restructure A Data File
                           9)   Program Generator
                          10)   Inquire

                          ##    Enter Your Choice From Options Above
```

LESSON 1
Using SENSIBLE SOLUTION for Data Entry and Update

We'll start by showing you some of SENSIBLE SOLUTION's capabilities. Included in your data files is a mailing-list manager that we'll use to demonstrate how SENSIBLE SOLUTION applications work.

Incidentally, the mailing-list manager is not just a toy demonstration program; it's a functional system you can put to good use. Once you've become familiar with SENSIBLE SOLUTION, you can tailor and expand it to fit your exact needs. That's one of the most important features of the SENSIBLE SOLUTION: you can build on previous work, without having to re-design from scratch.

Boot your computer (that's computerese for "turn on the power and re-start the operating system") then type

```
A> SENSIBLE
```

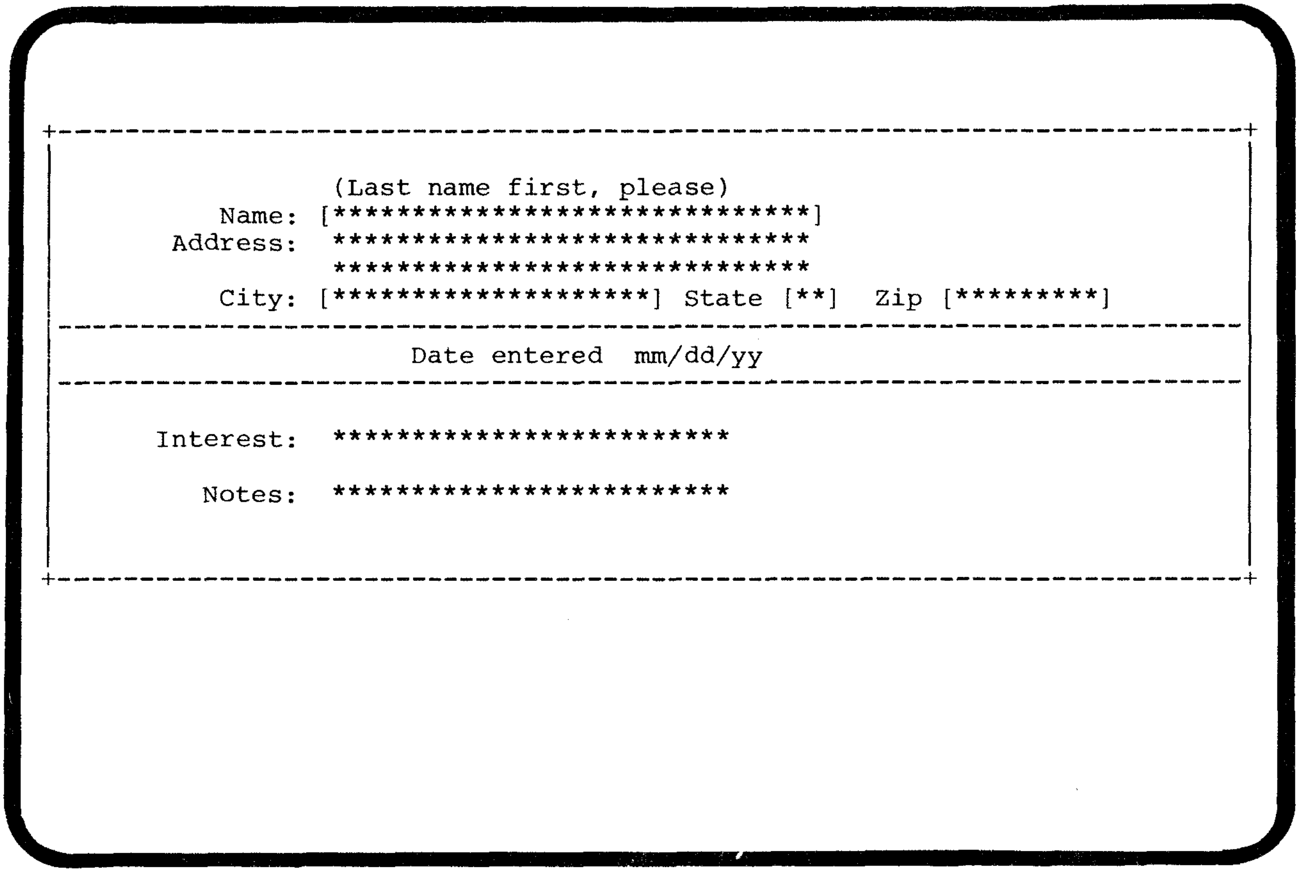
and press the carriage return key (on some terminals, the key is marked **[ENTER]** or **[RETURN]**). After viewing the copyright message, press **[RETURN]**. Your terminal display will clear and display the menu shown on the facing page. The SENSIBLE SOLUTION will always offer you a menu of choices or tell you exactly what it wants to know. This very important feature insulates you from typing and syntax errors and makes it unnecessary for the ultimate user (who may not be a programmer) to learn "computerese".

We're going to execute a SENSIBLE SOLUTION program, so type **1** and press **[RETURN]**. The SENSIBLE SOLUTION will clear the screen, load the command executive program, and ask you for the name of the .RUN file you want to execute. Press **[SPACE BAR]** and type **MAILLIST**.

Your terminal now displays the screen shown on the next page. Using this screen, you can enter new mailing-list records, modify or update existing entries, remove obsolete ones, and scan through the entire mailing list to find records by using several different **indexing keys:** last name, zipcode, city, or state.

A brief discussion is in order here to give you an overview of what a SENSIBLE SOLUTION program can do, and how you can control it. Following this summary, we will take you step by step through the selection and use of your screen controls.

Executing the program MAIlLIST --



```
+------------------------------------------------------------------------------+
|                                                                              |
|                   (Last name first, please)                                  |
|            Name: [******************************]                            |
|         Address:  ******************************                             |
|                   ******************************                             |
|            City: [********************] State [**]  Zip [*********]          |
|------------------------------------------------------------------------------|
|                        Date entered  mm/dd/yy                                |
|------------------------------------------------------------------------------|
|                                                                              |
|        Interest:  *************************                                  |
|                                                                              |
|           Notes:  *************************                                  |
|                                                                              |
|                                                                              |
+------------------------------------------------------------------------------+
```

Look at the screen display on the opposite page.

Every place on a screen display that you see an asterisk (*), a number sign (#), or a date (mm/dd/yy) is called a "field window". The field windows that appear on your screen allow you to enter data into a particular field in a file or to view the data that is already there. Each screen that the system displays contains one record, or line, of file information. Therefore, when you are viewing a display screen, you are actually looking at the contents of a record within one of the system's disk files.

To enter data into the system all you have to do is move the cursor to the appropriate field window and type in the data. When you have finished entering data onto the screen, you can then choose to save the "screen full" of data (a record) or to edit the screen. If you decide to save the screen of information, the system will transfer the information from the screen to the appropriate disk file where it will be saved. So how do you control all of this? Easy! The SENSIBLE SOLUTION screen menus and the keys on your computer terminal keyboard.

Screen Controls:

By pressing the escape key **[ESC]** you elicit the following display at the bottom edge of your terminal:

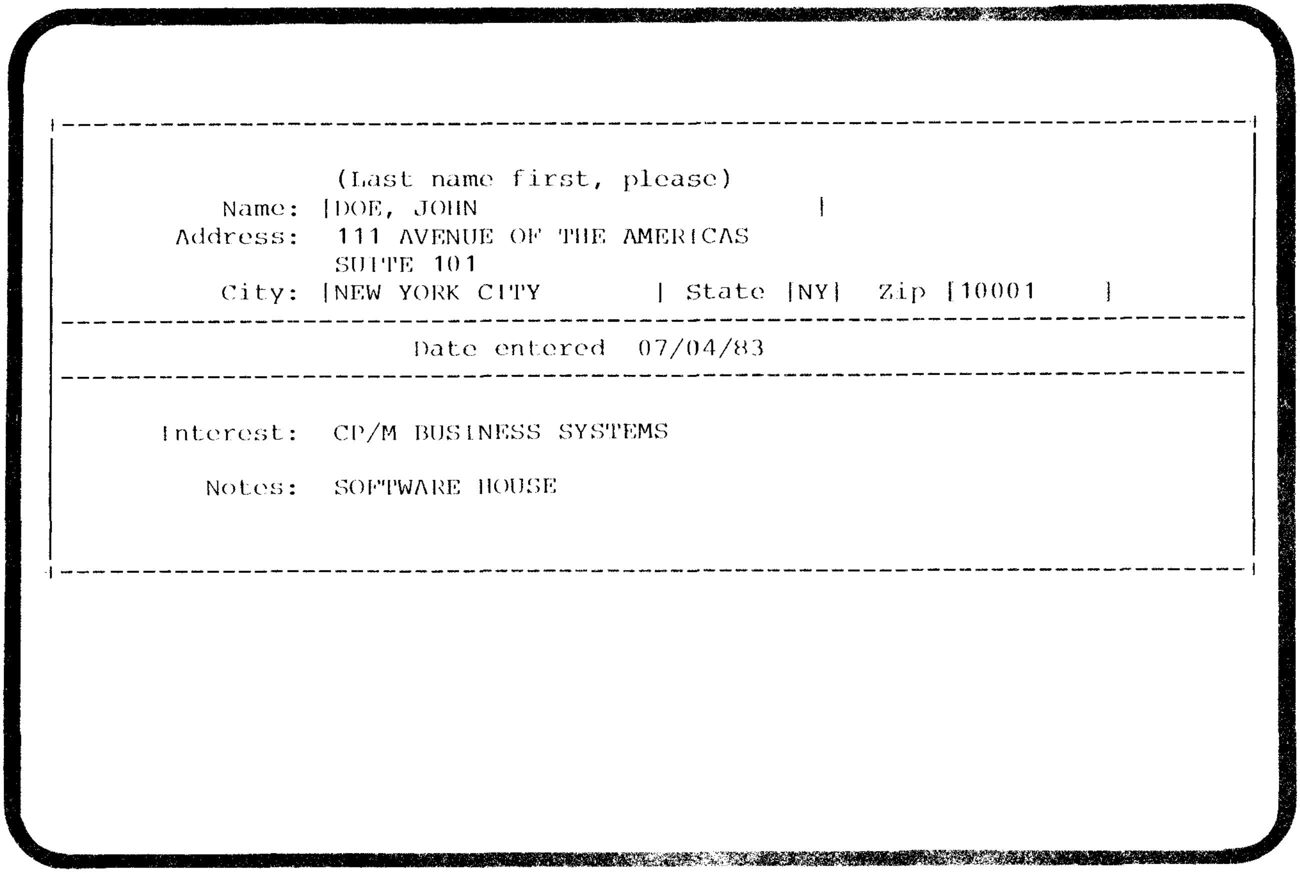
```
Begin  Next  Previous  End  Find  Save record  Remove record
Clear fields  Jump screen  Quit screen  Trace  ?[Help)
```

The word **Begin** is highlighted. By pressing your **[SPACE]** bar or **[RIGHT ARROW]** key, the highlight will move right, to the next option available through the displayed menu. The highlight indicates the option you wish to choose. Pressing the **[BACK SPACE]** or **[LEFT ARROW]** key will move the highlight back to the left. When you have highlighted the option you want, press your **[RETURN]** key and the choice will be activated.

Another way to activate a screen control is to press the character key indicated by the upper case character beginning each option. For example, press **[R]** to select the "Remove record" function. Throughout this manual brackets will be used to indicate that we are referring to a specific **key** on your keyboard.

Pressing **[ESC]** will always redisplay your control options menu. This allows

Press **[ESC] [B]** to find the beginning record --



```
|------------------------------------------------------------------------|
|                                                                        |
|                 (Last name first, please)                              |
|        Name: |DOE, JOHN                    |                          |
|     Address:  111 AVENUE OF THE AMERICAS                               |
|               SUITE 101                                                |
|        City: |NEW YORK CITY      | State |NY|  Zip |10001    |          |
|------------------------------------------------------------------------|
|                     Date entered  07/04/83                             |
|------------------------------------------------------------------------|
|                                                                        |
|    Interest:  CP/M BUSINESS SYSTEMS                                    |
|                                                                        |
|       Notes:  SOFTWARE HOUSE                                           |
|                                                                        |
|                                                                        |
|------------------------------------------------------------------------|
```

you to make a selection by either entering a single character to activate a function, or moving the highlight to the appropriate selection and pressing **[RETURN]**. For example, the option to "Clear fields" may be selected by pressing **[ESC]** (which calls up the menu and indicates to the computer that you are ready to make a selection), and then pressing **[C]** (the key for the letter C on your keyboard.) Or, you may move the highlight to the appropriate selection, "Clear fields" and press **[RETURN]**.

The control key, usually marked **[CTRL]**, does not work like the other keys on your computer terminal. When you press it, it doesn't produce a character on your display screen. Instead, it acts rather like the **[SHIFT]** key, but instead of changing the letter to upper-case it changes it to a special code that the computer recognizes as a signal to do some particular operation. We abbreviate the code by simply writing **[^D]**. In the future, any time you see **[^D]** written in this manual, hold the **[CTRL]** key down while you press the **[D]** key. In this example, your action would send a command to the system to delete a space or character from a field.

There are 20 different screen controls recognized by the system. By pressing **[ESC]**, 12 menu options are made available to you. Two of these options, **[ESC]** **[T]** AND **[ESC]** **[?]** are program assistance screens. Additionally, there are 7 commands for editing fields. These involve the use of the **[ARROW]** keys and the **[CTRL]** key. The remaining screen controls are used in transferring screens.

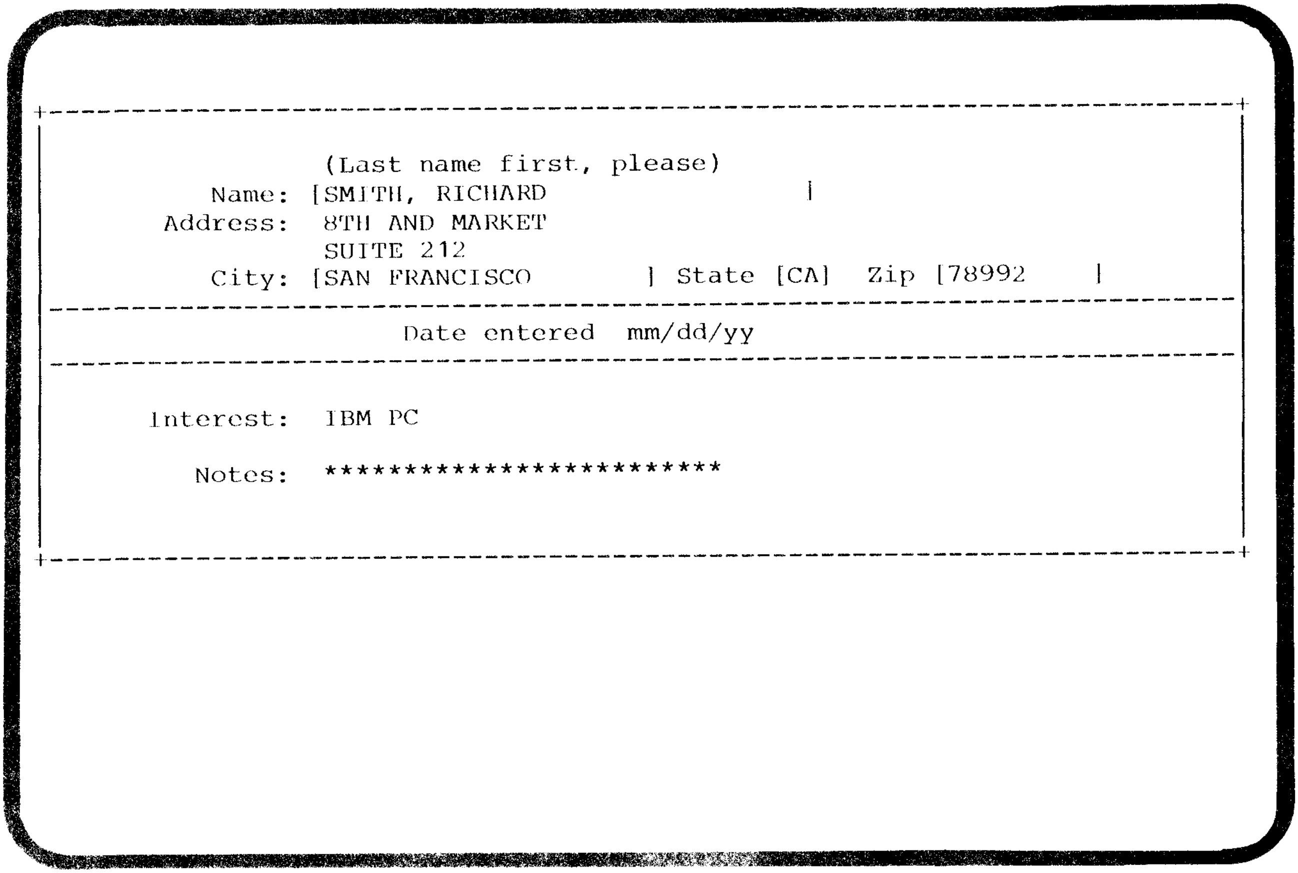
Now, using the control keys, we're going to look at the mailing list data file supplied with your SENSIBLE SOLUTION diskette. Then we will remove one of the entries from the mailing list data file, update another, correct a third, and add a few new entries.

We'll begin by looking for the first name in the list. Your cursor is located at NAME; so we'll be searching the list by name. Press **[ESC]** to display the screen controls menu, then press **[B] (find BEGINning record).** The computer will respond by finding **Doe, John** and displaying his record, as you see on the opposite page.

Now press **[N] (find NEXT record).** The entry for **JOHNSON, RICHARD** replaces **DOE, JOHN.** Press **[N] (find NEXT record)** again and again, and the computer steps through **SMIITH, ROBERT, SMITH, RICHARD,** and **SMYTHE, WINSTON.** Finally, the computer reports **"Search reached end of keys"** when you have exhausted the entries in the mailing list.

Because the computer printed an error message on your screen, the controls menu is no longer displayed. Normally, pressing **[ESC]** calls a menu to the screen. However, if you press **[ESC]** following an error message display, you will activate the Trace function. (The "Trace" lists the specific program

Using the **[ESC] [F] Find record** keys --



```
+---------------------------------------------------------------------+
|                                                                     |
|               (Last name first, please)                             |
|         Name: [SMITH, RICHARD                  ]                    |
|      Address:  8TH AND MARKET                                       |
|                SUITE 212                                            |
|         City: [SAN FRANCISCO       ] State [CA]  Zip [78992    ]    |
|---------------------------------------------------------------------|
|                     Date entered  mm/dd/yy                          |
|---------------------------------------------------------------------|
|                                                                     |
|     Interest:  IBM PC                                               |
|                                                                     |
|        Notes:  *************************                            |
|                                                                     |
|                                                                     |
+---------------------------------------------------------------------+
```

NOTE -- A field window surrounded by brackets indicates that the field has been designated as a "key field." In the above example, "Name" has been designated as a key field while the field "Interest" has not.

command line that the computer was executing at the time of error.) Press **[ESC]** again and the controls menu will appear. Too much work? Following the error message, you can eliminate passing through the "Trace" function by pressing **[RETURN]** or any key to redisplay the menu.

Just remember, as long as the screen control menu is diplayed on your terminal you may exercise any control option you wish, by pressing the keyboard key representing the upper case character lodged in each option or, by highlighting the option of your choice (using the **[BACK SPACE]** key or [SPACE BAR]) and pressing **[RETURN]**.

Press **[P] (find PREVIOUS record)** repeatedly and you can scan the entries in reverse alphabetical order until you have **Search reached beginning of keys.** Similarly, **[B] (find BEGINning record)** and **[E] (find ENDing record)** will display the first and last entries in the list.

Now we'll look for a particular name. First, press the **[C] (CLEAR fields)** key to blank out the "name" field window. The computer indicates that the field window is empty by filling it with asterisks showing you how many characters can be entered. If we had not cleared out the name field, the computer would have used the name currently displayed as a field index key and we'd never see a different record.

As you may have guessed by now, there are hardware keyboard keys and there are software "indexing keys", sometimes referred to as "key fields". Learning all these new terms will require some patience on your part. Let's continue through the MAILLIST example and you'll begin to see the relationship between files, records, fields, and key fields.

Type **SMITH** but don't press **[RETURN]**! Instead, press the **[ESC]** **[F]** **(FIND record)** keys, and the computer will display Richard Smith's record. The search finds the first record that matches the indexing key value you typed in and displays that information on the screen.

When the value you supply does not fill the entire field, **[ESC]** **[F]** **(FIND record)** uses the value as a **partial key** from which to start the search. Incidentally, any data left over also becomes part of the key. If you had not cleared the name field, but had typed "SMITH" over the top of "DOE, JOHN", the computer would have begun to search with "SMITHJOHN".

Now, find "SM". You should come up with "SMIITH, ROBERT" (did you clear the name field first?). We're going to correct the double-I in the name then save the corrected entry. Press **[ESC]**, which turns off the screen control menu and returns the cursor to the field window we have been searching. Press the **[RIGHT ARROW] (Cursor Right)** key twice, to position the cursor over the first "I". Now press **[^D] (Delete Character).** The first "I"

Execution Control Menu Keys -- The [**ESCAPE**] key turns the control menu on and off.

Begin End Find Next Previous Save record Remove record
Clear fields Jump screen Quit screen Trace ?(Help)

With the menu on, move the highlight to the selection of your choice and press [RETURN] or, enter the single capitalized character lodged in the option you prefer.

[B] Begin Finds first record (lowest value) in the file based on the field in which cursor appears.

[E] End Finds last record (highest value) in the file based on field in which cursor appears.

[F] Find Finds the record, which contains the field value, that most closely matches the displayed field value. Possible errors: "not a key field", "end of file encountered (record not found)"

[N] Next Finds next record in file. Possible error: "end of file encountered."

[P] Previous Finds previous record. Possible error: "beginning of file encountered."

[S] Save record Stores the displayed screen record in the disk file.

[R] Remove record Erases the displayed screen record from the disk file.

[C] Clear fields Clears all fields on screen to spaces.

[J] Jump screen Display next screen.

[Q] Quit screen Display previous screen.

[T] Trace Invokes the program debugging option.

[?] (Help) Display help screen.

NOTE: "find Next" and "find Previous" must be preceded by a "Find", "find Beginning," or "find Ending" in the same field. All searches trigger the "Relates" Trap command.

will disappear and the rest of the name slides to the left to fill in the deleted character space.

We've corrected the entry on-screen, but we haven't stored the correction on our disk-file yet. Press **[ESC] [S] (SAVE screen record)** and the computer will ask **SAVE this record? (Y/N).** This gives you a chance to check your entry and change it if necessary. Press the **[N]** key (No) to **not** save the record; the cursor will reposition at the name field.

Press **[ESC] [S] (SAVE screen record)** again and this time answer **[Y]** (yes). The computer will store the new information in place of the old. The "SMIITH" index key has disappeared, and the new "SMITH, ROBERT" key **follows** "SMITH, RICHARD" in the index. The SENSIBLE SOLUTION always keeps the indexes in alphabetical order even when several keys are changed at once.

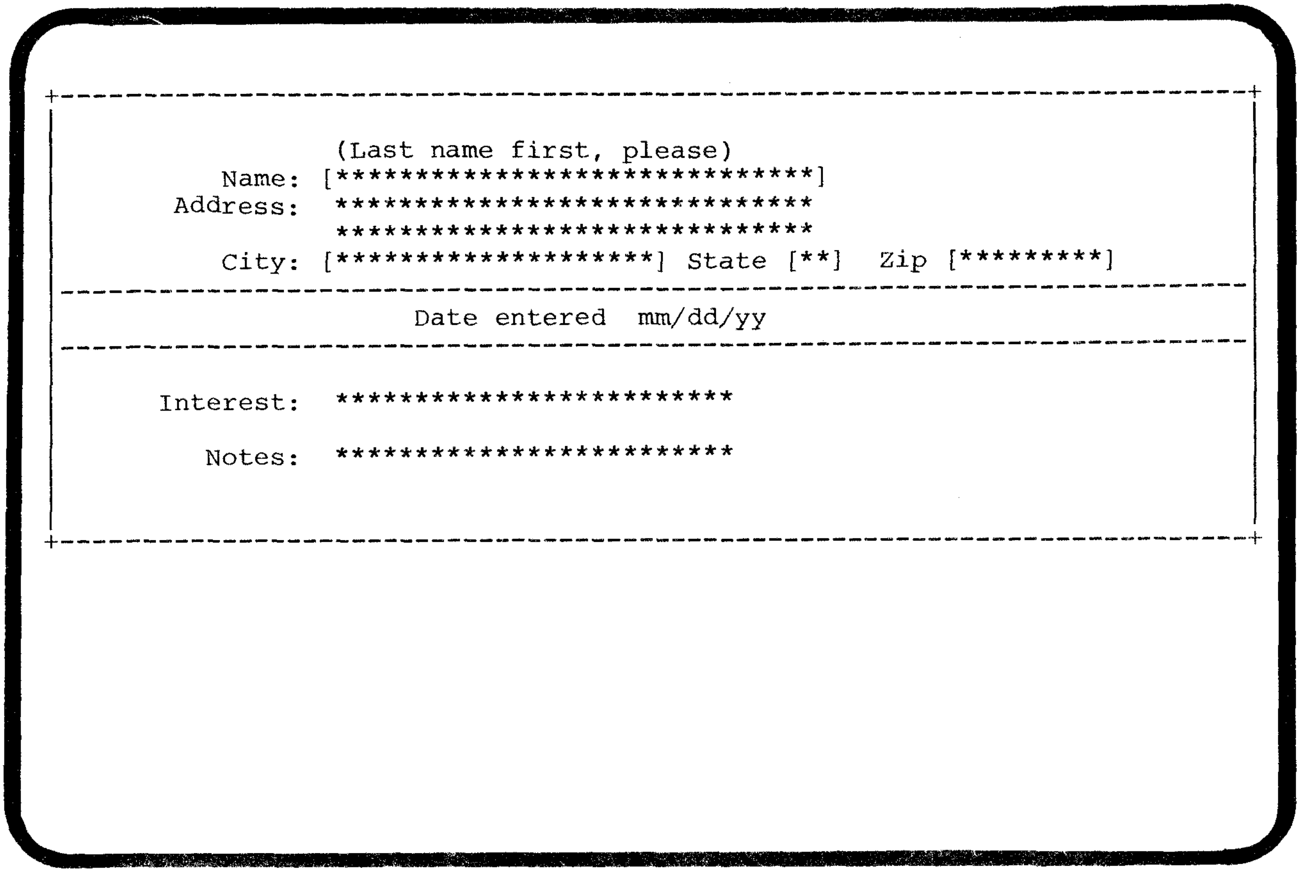
Information may be indexed by up to nine different **keys** for each data file. This mailing list is indexed by four keys: name, city, state, and zipcode. We indicate that a field is a **key** by putting brackets around the field window space defined for it on the terminal display (e.g., [key field]). Let's scan through the entries by zipcode.

The **(Previous Field) [UP ARROW]** and **(Next Field) [DOWN ARROW]** keys are used to move from field window to field window. Move through the field windows using the **(Next Field) [DOWN ARROW]** key until your cursor is positioned at the **zipcode** field. Now press **[ESC] [B] (find BEGINning record).** The computer displays the lowest zipcode on file along with the rest of the information in the entry. Scanning the address with **(find NEXT record) [ESC] [N]** will show the mailing list in zipcode order. Now look at the names; they are no longer in alphabetical order.

The SENSIBLE SOLUTION knows whether your cursor is positioned in a key field or not. In a field that is not an indexing key, the execution control menu selections presented when you press **ESC]** do not include the **Begin, End,** and **Find** options.

There is one other feature concerning record searches that we should talk about. Once you have done a record search using [ESC] [F] in a key field window, the system program considers that particular key field to be "set" -- the system will use that key field as the designated field until you move the cursor to another key field and conduct another search with [ESC] [F]. Actually, this feature works out rather nicely because once you move the cursor out of the "set" key field window, you can still page through the records by simply using [ESC] [P] or [ESC] [N]. You don't have to move the cursor back to the "set" key field just to look at the next ([ESC] [N]) or previous ([ESC] [P]) records. Experiment with these features on your system

To add a new record, first press **[ESC]** **[C]** (CLEAR fields) to clear all of the field windows on the screen --



```
+-----------------------------------------------------------------------+
|                                                                       |
|                (Last name first, please)                              |
|       Name: [******************************]                         |
|    Address:  ******************************                          |
|              ******************************                          |
|       City: [********************] State [**]  Zip [*********]      |
|-----------------------------------------------------------------------|
|                     Date entered  mm/dd/yy                           |
|-----------------------------------------------------------------------|
|                                                                       |
|   Interest:  *************************                               |
|                                                                       |
|      Notes:  *************************                               |
|                                                                       |
+-----------------------------------------------------------------------+
```

and you'll appreciate how easy it is to locate the exact record you want.

In this list the address lines, dates, and notes, are not keys. If you decide to modify this mailing list system later for your own use, you may want to change or add indexing keys. The SENSIBLE SOLUTION makes it easy to change fields and keys and automatically rebuilds your data files to accommodate the changes. It's all explained in the Reference Manual under Menu Selections 8) Restructure a Data File.

So far, we've been viewing or modifying information already in the list. Now let's add a new record. First, press the **(CLEAR fields) [ESC] [C]** keys. The computer will leave the field labels on-screen but fills the field where the cursor is with asterisks to denote that it is empty of information.

With the cursor located at the **name** field, type in the name of one of your customers (last name first, so the indexing key can find the surname). If the name fills the entire field, the cursor automatically jumps to the next field window **"Address"**; otherwise, press **[RETURN]** to tell the computer that the field has been completely entered.

Continue down the screen entering address, city, state, etc.. You can skip a field (leave it blank) by using the **(Next Field) [DOWN ARROW]** key. **(Previous Field) [UP ARROW]** will let you go back to correct a previous field.

You may edit the contents of a field with the **(DELETE CHARACTER) [^D]** and **(INSERT CHARACTER) [^I]** keys. **(INSERT CHARACTER) [^I]** bumps everything in the field one space to the right to make room for a new character under the cursor. Characters at the right end of the field fall off and are lost. Pressing **[^U]** will clear the field window where the cursor is positioned. It clears **only** the entry in that field window, other fields on the screen are not disturbed.

You can save the record even if you haven't filled in all the fields. Just press **[ESC] [S] (SAVE record)** and the computer will ask you to confirm **SAVE this record? (Y/N).** Take a look at the entry. If it's right, press **[Y]**. The computer will save the new record and insert all the indexing keys in their proper order.

To delete an existing record, bring the entry up on the screen and press **(REMOVE record) [ESC] [R].** The computer asks you to confirm **REMOVE this record? (Y/N).** Once a computer has erased data, it's gone forever; so this confirmation insures against accidental loss of information due to an occasional typing error.

We know how to scan the information, edit it, add new data, and delete old

Cursor Screen Controls --

These controls operate when the execution control menu is off. They allow you to move your cursor from field to field, and to edit your data entry.

[UP ARROW] = Move cursor to beginning of previous field. (Locks at topmost field)

[DOWN ARROW] = Move cursor to beginning of next field. (Locks at lowest field)

[RIGHT ARROW] Move cursor one position right within field. Value of the field is unchanged.

[LEFT ARROW] = Move cursor one position left within field. Value of the field is unchanged.

[^D] = Deletes the character "under the cursor". Remaining characters in field field shift left to fill, and a blank appears at right end of field.

[^I] = Inserts a blank "under the cursor". Remaining characters in field shift right. Rightmost character in field is lost.

[^U] = Clears the displayed value from the field window.

data. The control keys work the same way in every SENSIBLE SOLUTION application. There is one last control to learn, **(QUIT screen) [ESC] [Q].** Press these keys to finish working on the current screen and you will be returned to the previous screen used by the system, in this case, **The SENSIBLE SOLUTION** main menu.

One last point concerning screen controls! Let's say that you just put some long hours into data entry and now its quitting time. You can not simply turn off your computer in the middle of a program. It is imperative that you make an orderly exit from the system. Here's how and why:

> **If you do not follow an ORDERLY exit sequence prior to turning off your computer system, you risk seriously corrupting the integrity of your data. Every time that you choose to stop executing a SENSIBLE SOLUTION program (data entry and/or data retrieval) you must use [ESC] [Q] to exit to the previous screen. Repeat this process until you have exited all the way back to the operating system prompt. Once the operating system prompt is displayed on your screen, you may then turn off your computer system.**

In a minute, we're going to create a new SENSIBLE SOLUTION application from scratch to show you how programming is done in the SENSIBLE SOLUTION. But first, let's clear up a little confusion. We've quietly slipped some "computerese" into the discussion, and it's time to explain what it means. We're talking about **files, records** and **fields.**

Most people know that computers deal in 1's and 0's, **binary digits** (abbreviated as **bits**). Microcomputers handle data in chunks of eight bits at a time **-- bytes.** Each byte can represent an arithmetic quantity, a character, or some other type of data. The computer program determines how each byte will be interpreted.

The actual nuts and bolts of binary data representation are "invisible" to the computer programmer. Programmers usually think in terms of numbers and characters being manipulated directly by the program just as you consider "$5,000.00" to be a money amount not a string of alphanumeric characters. Actually, you see it as an amount when you look at it as "how much?" and as characters when you're typing it into a column of figures. Your program lets you switch between the two interpretations when appropriate.

Information is usually gathered into groups of related data. Consider a file folder full of invoices. The folder is a **file** and it has a **name:** the label on the folder. Inside is a sheaf of pages, one per transaction. Each page is a **record** of the transaction. Finally, each page consists of a number of different items: customer name, address, invoice number, date, Files, records, fields, and key fields --

```
 FILE
  ||
  ||
 \  /
  \/
MAILLIST
```

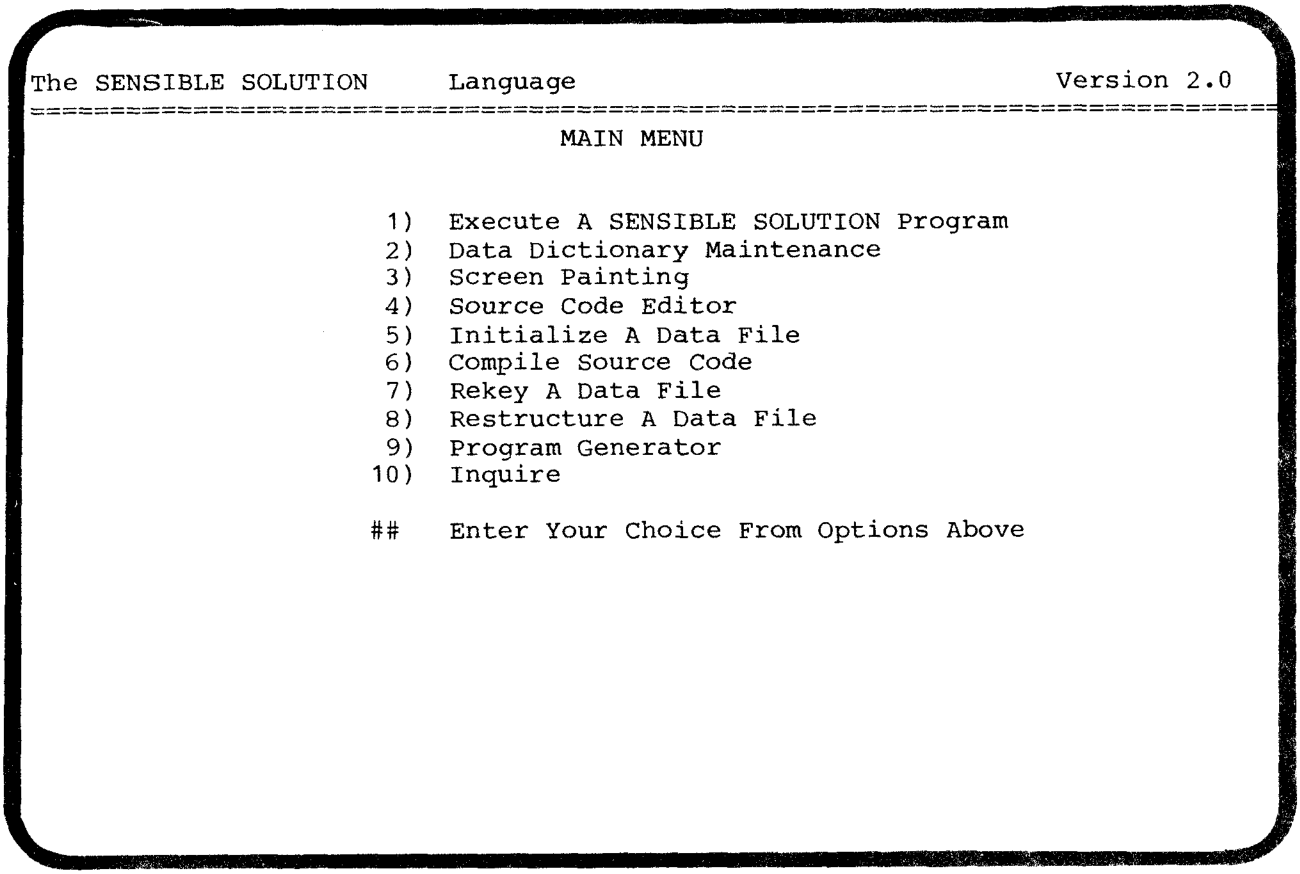
<---------------------------------fields--------------------------------->

	KEY			KEY	KEY	KEY		
	Name	Address 1	Address 2	City	State	Zip	Interest	Notes
RECORD 1								
RECORD 2								
RECORD 3								
RECORD 4								
RECORD 5								
etc.								

part number, quantity, back order, and so on. In computer lingo, these elementary "particles" of data are called **fields.**

As you continue, you're going to be using fields a lot. They are the basic building blocks of data file management. Take some time to look at the picture on the opposite page and you will see what we mean by **files, records, fields, and key fields.**

After typing SENSIBLE **[RETURN]** --



```
The SENSIBLE SOLUTION     Language                              Version 2.0
==============================================================================
                                 MAIN MENU


                           1)   Execute A SENSIBLE SOLUTION Program
                           2)   Data Dictionary Maintenance
                           3)   Screen Painting
                           4)   Source Code Editor
                           5)   Initialize A Data File
                           6)   Compile Source Code
                           7)   Rekey A Data File
                           8)   Restructure A Data File
                           9)   Program Generator
                          10)   Inquire

                          ##    Enter Your Choice From Options Above
```

LESSON 2
Using the Program Generator

Now that you're familiar with data entry and update in the SENSIBLE SOLUTION, we're going to create an application system from scratch. This will demonstrate more capabilities of the SENSIBLE SOLUTION, particularly the power and ease of implementing an applications design.

Your display should be showing the SENSIBLE SOLUTION main menu. If you're still in the mailing list, press **[ESC]** **[Q]** now and wait for the menu. If you're at the operating system level, be sure that your SENSIBLE SOLUTION copy is in the drive. You did make a copy of the distribution diskette, didn't you? Good! Now type

```
     A>SENSIBLE
```

and press **[RETURN]**. The SENSIBLE SOLUTION main menu will appear on your screen.

We'll begin by "painting" a screen -- a terminal-display layout for data entry and update. After that, the SENSIBLE SOLUTION will automatically generate a simple program from it.

Select "Main Menu 3) Screen Painting" by pressing **[3]** and then **[RETURN]**.

The following display will appear on your screen:

```
                                               +---------------+
Enter the type of format you wish to load      |Screen format|   Reporter format
                                               +---------------+
```

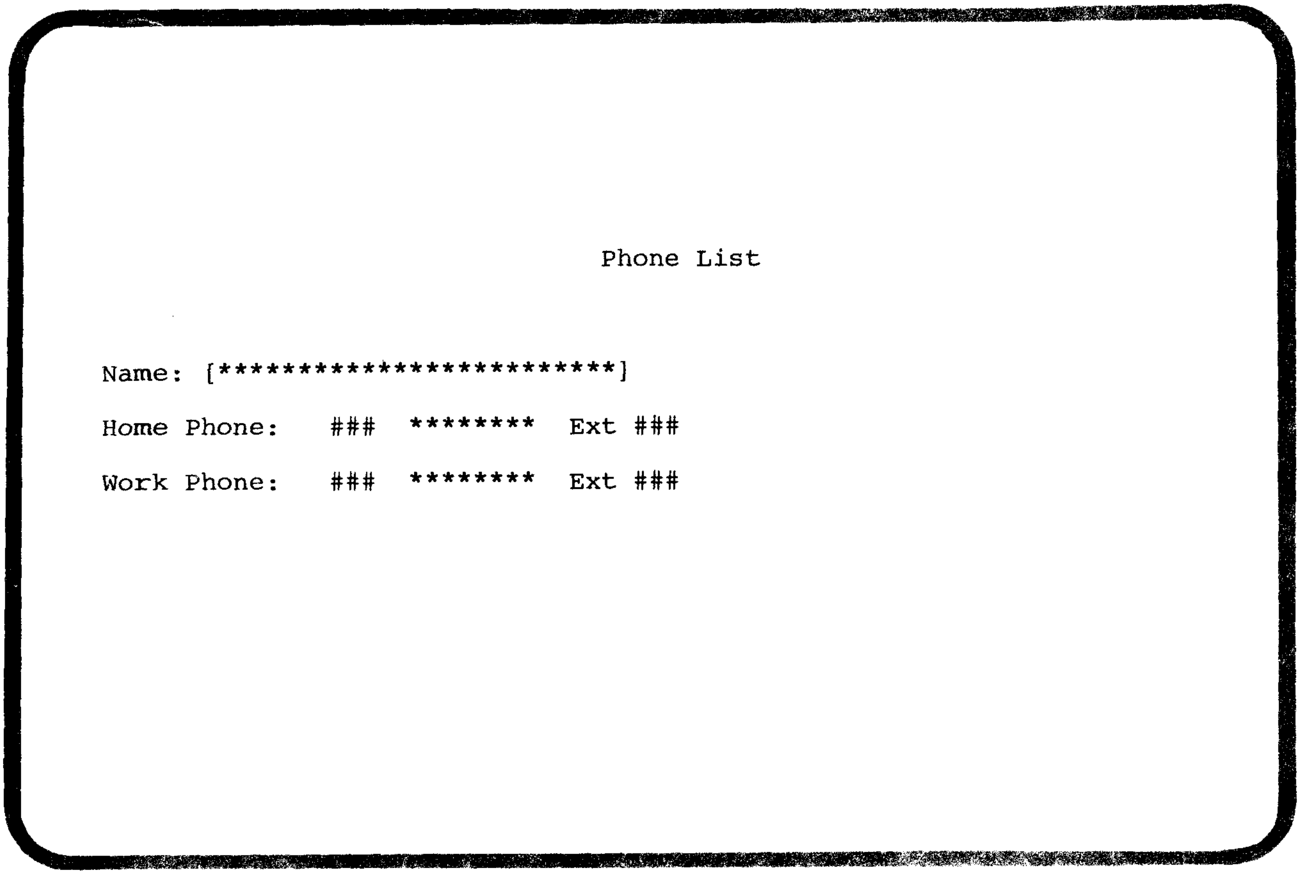
Press **[RETURN]**.

Now you will see:

```
               Enter the name of the screen format: ?:???????
```

Press your **[SPACE BAR]** and type **PHONLIST**.

Paint the screen headings on this page. Don't put in the *'s or #'s. Do include the brackets after the field label **"Name:"** -- they will remind you that "name" is a key field.



```
                                Phone List


  Name: [*************************]

  Home Phone:   ###  ********  Ext ###

  Work Phone:   ###  ********  Ext ###
```

The SENSIBLE SOLUTION will create all data and program files on the same disk drive that it resides on. You can, however, specify a different disk drive than the one you're on. If you had typed **B:PHONLIST** all files would be created on drive B. You'll find more information about file names in the operating system User's Guide that came with your computer.

The SENSIBLE SOLUTION Screen Painter will be creating a data file from this entry: **PHONLIST.SCC** The **".SCC"** portion of the file-name is called the **extention** and indicates the type of file -- in this case, a screen file.

To continue, now you will see the following question on your screen:

```
          +---+
New File? |Yes|  No
          +---+
```

Press **[RETURN]** and your screen will display a status line:

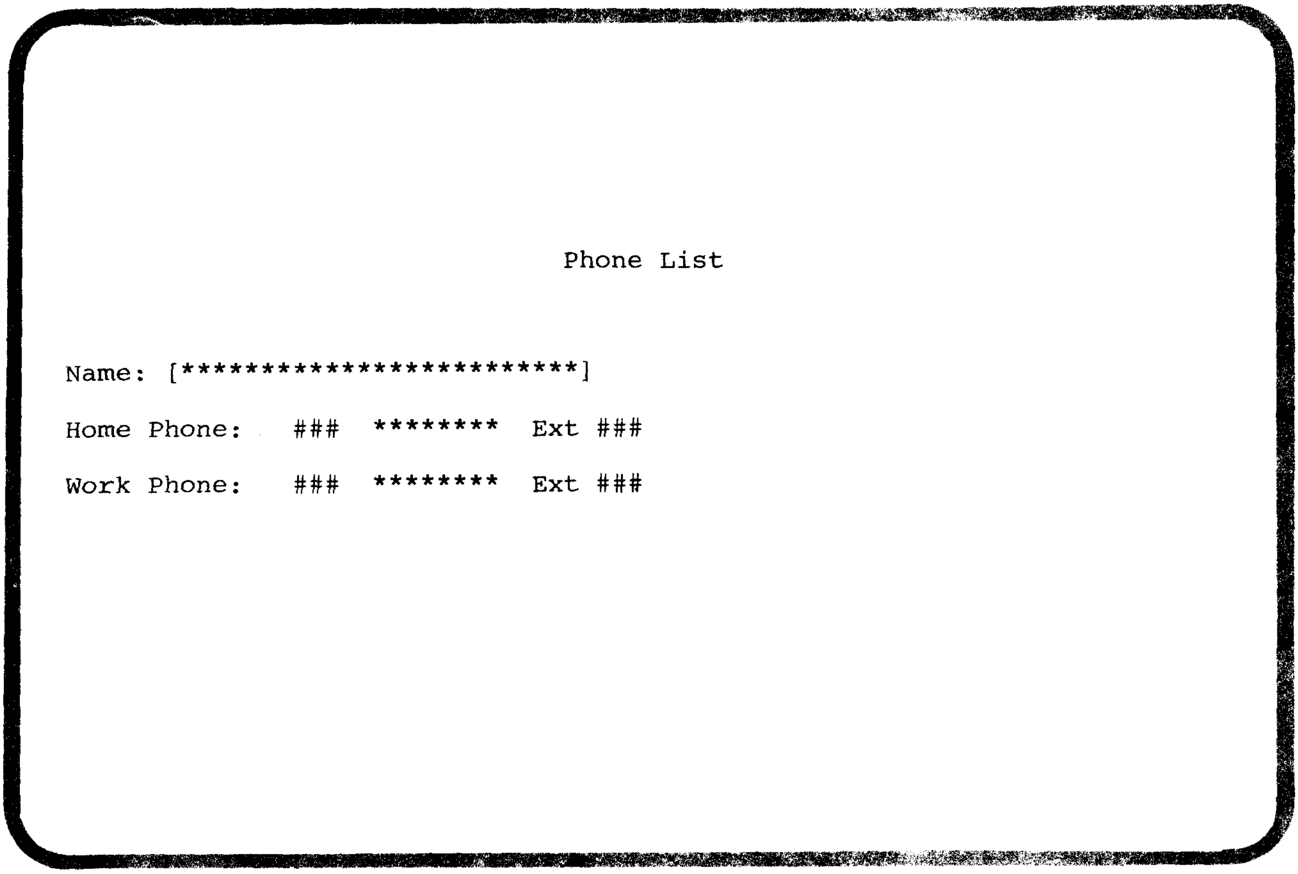
```
@:PHONLIST.SCC  file opened  col=001 row=01
```

First, we want to lay out the "picture frame" for our screen -- the marks and labels we want the screen to show indicating what is to be entered for each **field.** After this is done, we can define the fields themselves, which you can think of as "windows" through which data values will be displayed. During "screen painting", you are telling the computer how to display and manipulate values.

To begin, simply type what you want to see on the display at the desired positions. You'll find the cursor-positioning control keys useful: **(CURSOR UP) [UP ARROW], (CURSOR DOWN) [DOWN ARROW], (CURSOR LEFT) [LEFT ARROW]** and **(CURSOR RIGHT) [RIGHT ARROW].** These keys let you move your cursor anywhere on the display without disturbing information already displayed. Go ahead and "paint the screen" headings as shown on the facing page. Don't put in the *****'s or **#**'s.

(INSERT CHARACTER) [^I] puts a space "under the cursor" and bumps everything else on the cursor line to the right one position; the character at the right edge of the screen falls off the edge and disappears. **(DELETE CHARACTER) [^D]** works just the opposite; the character under the cursor disappears, the line moves to the left, and a space appears at the right edge of the screen.

Paint the screen headings on this page. Don't put in the ***'s** or **#'s**. Do include the brackets after the field label **"Name:"** -- they will remind you that "name" is a key field.



```
                               Phone List


Name: [*************************]

Home Phone:   ###  ********  Ext ###

Work Phone:   ###  ********  Ext ###
```

Similarly, **(ADD LINE) [ESC] [L]** and **(DELETE LINE) [ESC] [D]** insert and delete the entire horizontal line the cursor is on and bump all lines **below** that line down or up to compensate. Incidentally, the SENSIBLE SOLUTION will refuse to delete a line on which a field has been placed. You must remove the field first, **then** delete the line.

To create a box drawing on the screen, place the cursor at the desired position for the upper left corner and press **[ESC] [B] (BOX).** Now move the cursor to the desired position for the lower right corner of the box and press the **[SPACE BAR].** A box will appear on your screen. To create a **horizontal line** on your screen, simply create a box with no height. To create a **vertical line** on your screen, simply create a box with no width.

[ESC] [U] (UNBOX) is used to remove box drawings from the screen format. Place the cursor at the top, left corner of the box you wish to remove and press **[ESC] [U].**

With a little practice it becomes a snap to quickly generate very impressive screens and move labels back and forth until they are centered and neatly framed. Play with these editing control keys for a while to get the feel of designing a screen. You should end up with a display that looks like the screen shown on the facing page. Don't forget to leave room for the field windows!

Just as you learned in executing a SENSIBLE SOLUTION program, there is a menu to manipulate the display. Move your cursor to the right of the label **"Name:".** Place it at the starting point for your field window.

Now press **[ESC].** The Screen Painting menu will be displayed:

```
+--------+
|Add fld|   Remove fld  Move fld  Show fld  Field chg  file Chg  Hard copy
+--------+
                Quit  Del lne  add Lne  Box  Unbox  rEdisp scrn
```

When the menu appears, your cursor will move up to position itself over "Add field". Since we want to add a field, this is the menu function to select. Simply press **[RETURN]** and the display will request:

```
        Enter the name of the field ****************
```

Screen Painting Cursor Control Keys --

These controls let you move your cursor anywhere on the screen without disturbing information already displayed.

[UP ARROW] = Cursor up

[DOWN ARROW] = Cursor down

[RIGHT ARROW] = Cursor right

[LEFT ARROW] = Cursor left

[^D] = Deletes the character "under the cursor". Remaining characters in field shift left to fill, and a blank appears at right end of field.

[^I] = Inserts a blank "under the cursor". Remaining characters in field shift right. Rightmost character in field is lost.

Type in PHON.NAME and press **[RETURN]**. On the screen you will see the message:

The field entered was not found. Do you wish to create it? Y

The default answer **Yes** is activated when your press **[RETURN]**. Now you will see a list of specifications to use in defining the field you are creating. As you progress down the list, you will be prompted to answer questions about each of the specifications which apply to the field as you define it. Here we go!

Field name: Phon.Name
File name:
Field description:
Field type: (A,N,D,O,R)
Size:
Number of decimal places:
Offset:
Key: (Y/N)
Entry mask:
Upper case entry only: (Y/N)
Carriage return required: (Y/N)

PHON.NAME will display as the default entry for "Field name:".

Every SENSIBLE SOLUTION field must have its own unique name. Even if you use another data file, the field name cannot be duplicated. The computer searches the Data Dictionary to see if it already knows the name. If it does, the field is displayed as previously defined and the rest of the questions are skipped. This insures that all programs will handle data consistently.

Field names can be up to 15 characters long and may contain almost any character except for [ ] ( ) < > + - & * and /. These characters are used in calc expressions, which we'll be talking about later. This may come as a shock if you're used to BASIC-style field names that must start with a letter and contain only letters and numbers. For example, "1000.00", "MEMORY\1", and "FERD!IS@" are all valid **SENSIBLE SOLUTION** field names.

This gives you complete freedom to create names that are understandable to you. Obviously, things can get out of hand if you don't use a little discretion! We recommend you choose word names like "CUST.PYMT.RCVD" or "CUS.RCVBLSO1" to help you remember what the data represents and which file it's stored in.

Screen Painting Menu Control Keys --

The **[ESCAPE]** key turns the menu on and off.

Add fld   Remove fld   Show fld   Field chg   file Chg   Hard copy
Jump   Quit   Del lne   add Lne   Box   Unbox   rEdisp scrn

With the menu on, move the highlight to the selection of your choice and press **[RETURN]** or, enter the single capitalized character lodged in the option you prefer.

[A] Add fld Place field at cursor location.

[R] Remove fld Remove field at cursor location.

[S] Show fld Gives defined field name, file, size, location on screen, and whether field is a key.

[F] Field chg Allows redefinition of any element of field specifications.

[C] file Chg Allows redefinition of file.

[H] Hard copy Send a format description to a disk file or to the printer.

[J] Jump Will shift your display to the right or left so that you may view 127 columns of a **reporter** format with your 80 column CRT.

[D] Del lne The line below the cursor is deleted. All lines below move up and a blank line is created at the bottom of the screen.

[L] add Lne A blank line is inserted below the cursor. All lines below move down and the last line on the screen is lost.

[Q] Quit This ends an editing session.

[B] Box Create a box drawing on the screen.

[U] Unbox Remove box drawings from the screen format.

[E] rEdisp scrn Correctly redraws any boxes on the screen which may have been altered or disarranged due to you inserting or deleting spaces.

If you find you have mistyped the field name, you can re-enter the correct field name at this time. Press **[RETURN]** to continue down the list.

Enter File Name -- Every field must exist in a data file. Right now, we haven't told the SENSIBLE SOLUTION about **any** data file(s) for this screen. Type **PHONLIST** (you don't have to press **[RETURN]** the computer knows that **PHONLIST** is a full-length 8-character file name all by itself). Later, when we are defining more fields, the computer will offer **PHONLIST** as a default file name.

Now you will see:

This file does not exist. Do you want to create it? Y

Press **[RETURN]** and the data file will be created.

Field description: -- You may use this 30 character long space to enter a description or reminder of what purpose your field serves. Enter whatever remark or description you wish. This has no effect while executing programs. Your description will appear in the Data Dictionary. Press **[RETURN]**.

Enter the Field type:(A,N,D,O,R) -- Data fields may be:

Alphanumeric type: letters, digits or punctuation

Numeric type: digits, decimal point and minus sign only

Date type: automatically tested on entry for validity. You can add days to a date, subtract dates from each other, and key on dates.

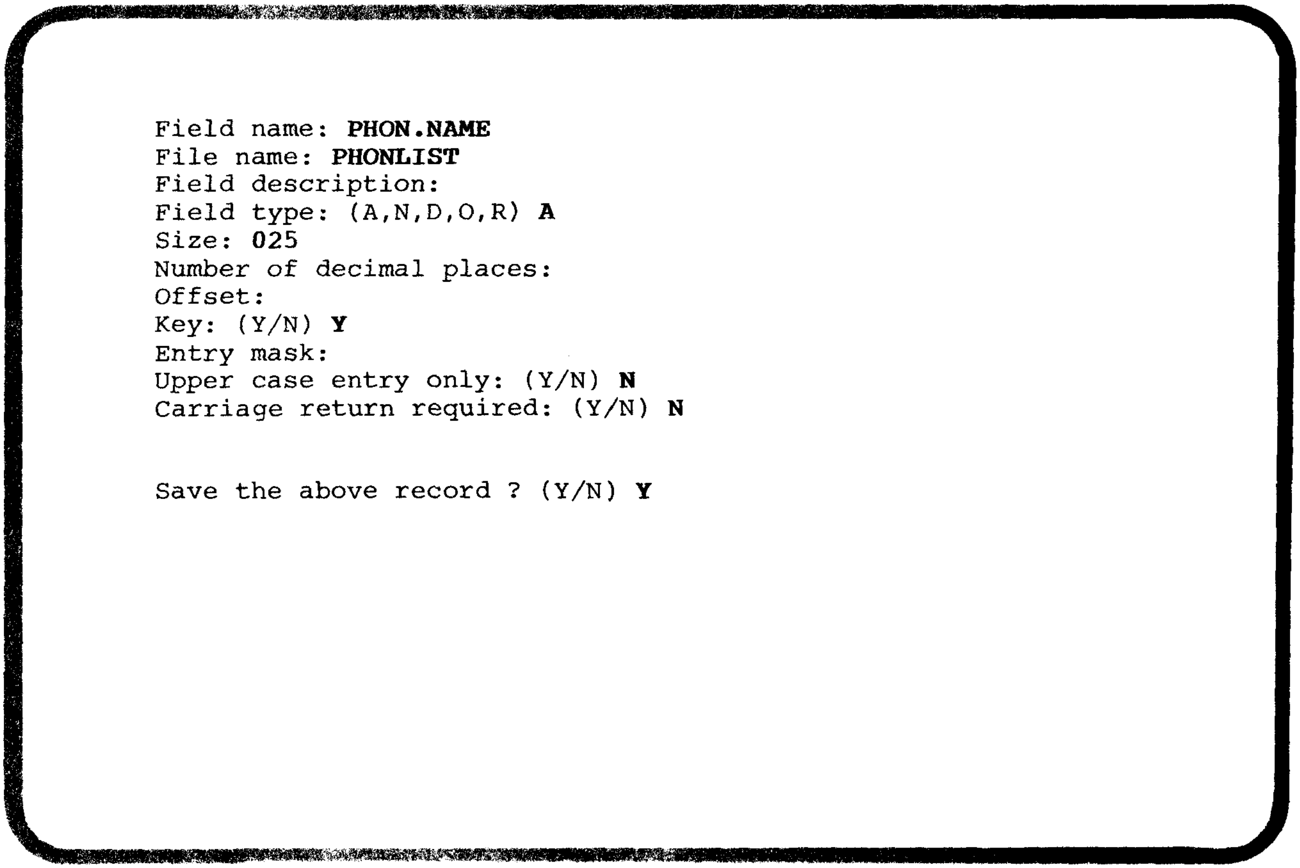
Overlay type: two fields can be defined as one--for instance if you have a field for "first Name" and a field for "last name" you can define an overlay field which contains "first and last name" known as "name."

Record number type: this field contains the record number for each data record saved. You may search for records by using this field.

Alphanumeric fields are the most common so **A** is offered as the default. We want **PHON.NAME** to be alphanumeric so press **[A]** or **[RETURN]**

Enter the Size--The number of character positions in the "window" for this field. Type 25 and press **[RETURN]**.

[ESC] [A] Add fld, adding the field PHON.NAME --



```
Field name: PHON.NAME
File name: PHONLIST
Field description:
Field type: (A,N,D,O,R) A
Size: 025
Number of decimal places:
Offset:
Key: (Y/N) Y
Entry mask:
Upper case entry only: (Y/N) N
Carriage return required: (Y/N) N


Save the above record ? (Y/N) Y
```

Note: You can use **[UP ARROW] and [DOWN ARROW]** to position your cursor next to any field specification you wish to alter. Press **[^U]** to clear the current definition and re-enter the specification you desire.

At any point during its creation, you may abort the field definition. Use your **[UP ARROW]** key to position the cursor in the **"Field name:"** field window at the top of the list of specifications. Press **[ESC] [Q]** and see the message.

The field entered was not found. Do you wish to create it? Y

Answer **[N]** (no). On the screen you will find that the cursor is returned to where it was last located before you chose to add a field. If you choose to escape, no field information will have been created, but the file name entered will be created. This escape feature will allow you to stop an entry if you've made a mistake.

Let's go ahead and create this field now. Press **[ESC] [A] (Add field).** Enter the field name: **PHON.NAME** and respond "yes" you do want to create it. Now enter the following:

File name:	PHONLIST	press **[RETURN]**
Field description:		press **[RETURN]**
Field type:	A	
Size:	25	press **[RETURN]**

Now you will find that the cursor skips down the list of definitions to **Key: (Y/N).**

Because you defined the field PHON.NAME as "alphanumeric," the question **"number of decimal places?"** does not apply. If you had defined your field as numeric you could specify as many as four places after the decimal point.

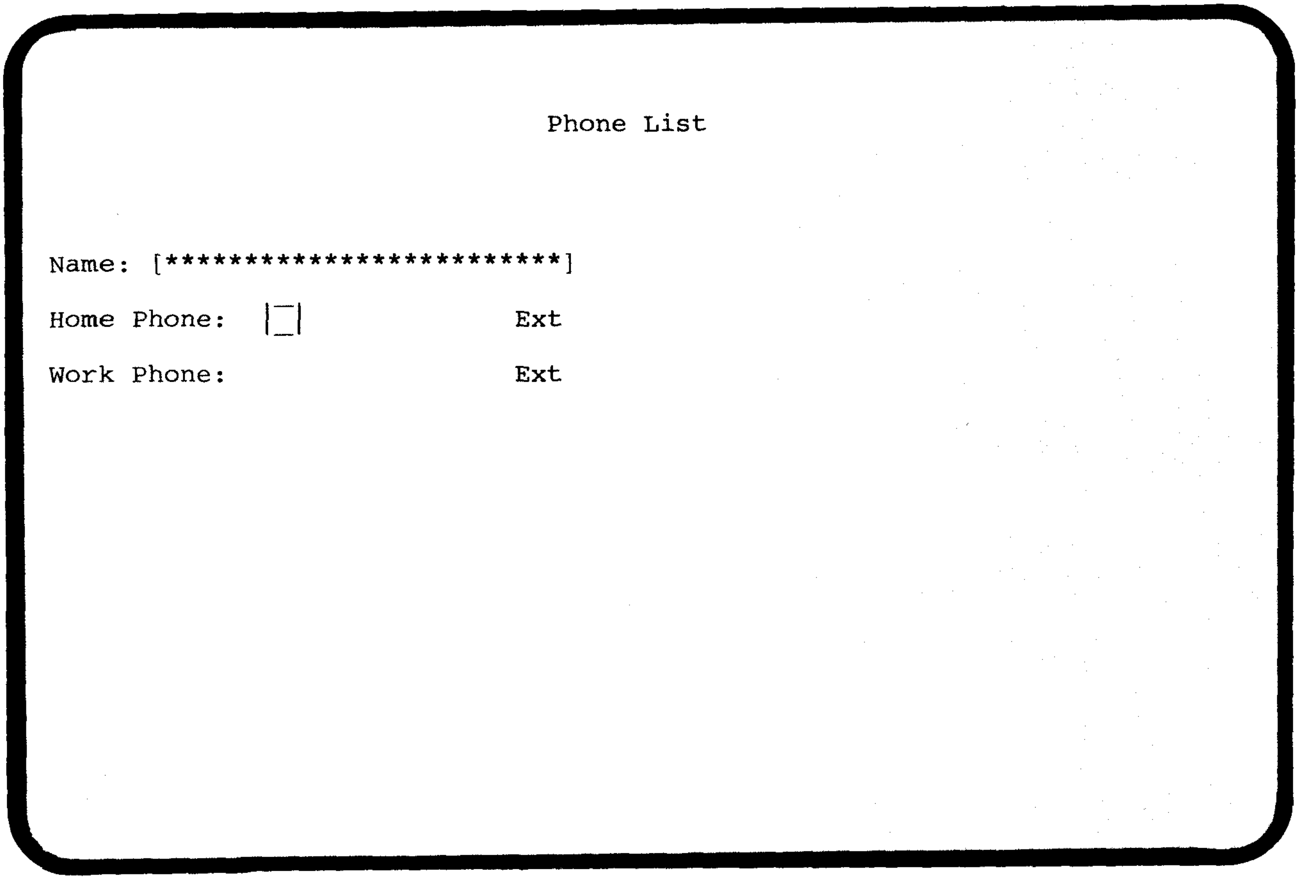
You will only enter information into **Offset** when you are defining a field as a type **O** -- overlay field.

Key (Y/N) -- File records may be **index-keyed** by as many as nine different fields. You don't want to slow the computer down by indexing on unnecessary keys. In this case, the name **is** a key we want to use in searching for data records so press **Y** -- no need to press **[RETURN].**

ENTRY MASK: -- an entry mask is the format of a field that is stored in the Data Dictionary. (See the section on Masking in the Reference Manual, Data Dictionary.) Because we do not want to define a mask for this field , press **[RETURN].**

Upper case entry only: (Y/N) -- If you answer yes, all data entry to this field will be automatically converted to upper case characters. Answer **N.**
Carriage return required: (Y/N) -- By answering **Y** you are specifying that any user who enters data to this field will be required to press **[RETURN]**

Use the arrow keys to move the cursor to the first space provided for "home phone" area code --



```
                                 Phone List


Name: [*************************]

Home Phone:  □              Ext

Work Phone:                 Ext
```

after each entry to get to the next input field. If you answer **N,** the system will automatically jump to the next input field if you completely fill this field with characters. If it is not filled, you will need to press **[RETURN]** to get to the next input field. For our purposes, answer **N.**

If you find that you have made an error at some point in the creation of this field definition, you may use your **[UP ARROW]** and **[DOWN ARROW]** keys to move your cursor to the erroneous entry and make any necessary corrections. Press **[^U]** to clear the current definition, then re-enter the specification you desire.

At this point you will be asked:

Save the above record? (Y/N) Y

Press **[Y]** or **[RETURN].** The **SENSIBLE SOLUTION** now knows exactly how you want this file represented and places 25 *'s next to the "Name:" label on your display to indicate that the screen will accept up to 25 characters in that "window". The field is added to your screen and the cursor will return to the first character of the field window. Congratulations! You have just added your first field. Now let's add the rest of the fields.

Using the arrow keys, move the cursor to the first space provided for your "home phone" (area code). Press **[ESC]** to call up the Screen Painting menu.

We want to add a field, and the **Add field** option is highlighted, so press **[RETURN].**

Following the procedure described above, enter the following information:

LABEL	**QUESTION**	**ANSWER**	**NOTES**
Home Phone:	Field name:	PHON.AREAHOME	(area code)
	File:	PHONLIST	default value--
	Field type:	N	**press [RETURN]**
	Size:	5	
	Number of decimal places:	0	
	Key:	N	
	Entry mask:	\(###\)	
	Save?	Y	

Because it is not necessary to enter a "Field Description," and we do not want to require "Upper case entry only" or "Carriage return," these questions were omitted from the above list. Simply press **[RETURN]** or **[DOWN ARROW]** accepting the default answers.

SENSIBLE SOLUTION File Extensions --

File extensions created and used with SENSIBLE SOLUTION:

*.SCC	Screen or Reporter Format files
*.SRR	Command Source Code files
*.RUN	Compiled Command files
*.IQ	Inquire Format files
*.LST	Screen/Reporter Format or Source Code ASCII files
*.MS	Master Data file
*.KS	Key file (for the .MS file)

Operating System file extensions used with SENSIBLE SOLUTION:

*.COM	Compiled executible program - CP/M, MP/M
*.EXE	Compiled executible program - MS DOS, PC DOS

Did you notice that the entry mask that you defined for this field was not immediately reproduced on your screen. The 5 # signs indicate the number of spaces reserved for this field. The parentheses will show up after we compile the program.

Now position your cursor after the home area code field and press **[ESC]** and **[RETURN]** to add the home phone number field.

Home Phone	Field Name:	PHON.NUMHOME	(home phone number)
	File:	PHONLIST	**press [RETURN]**
	Field type:	A	
	Size:	8	
	Key:	N	
	Entry mask:	###-####	
	Save?	Y	

Position your cursor after the home phone number field, then press **[ESC]** and **[RETURN]** to add a home extension field.

Home Phone	Field name:	PHON.EXTHOME	(home extension)
	File:	PHONLIST	**press [RETURN]**
	Field type:	N	
	Size:	3	
	Number of decimal places:	0	
	Key:	N	
	Save:	Y	

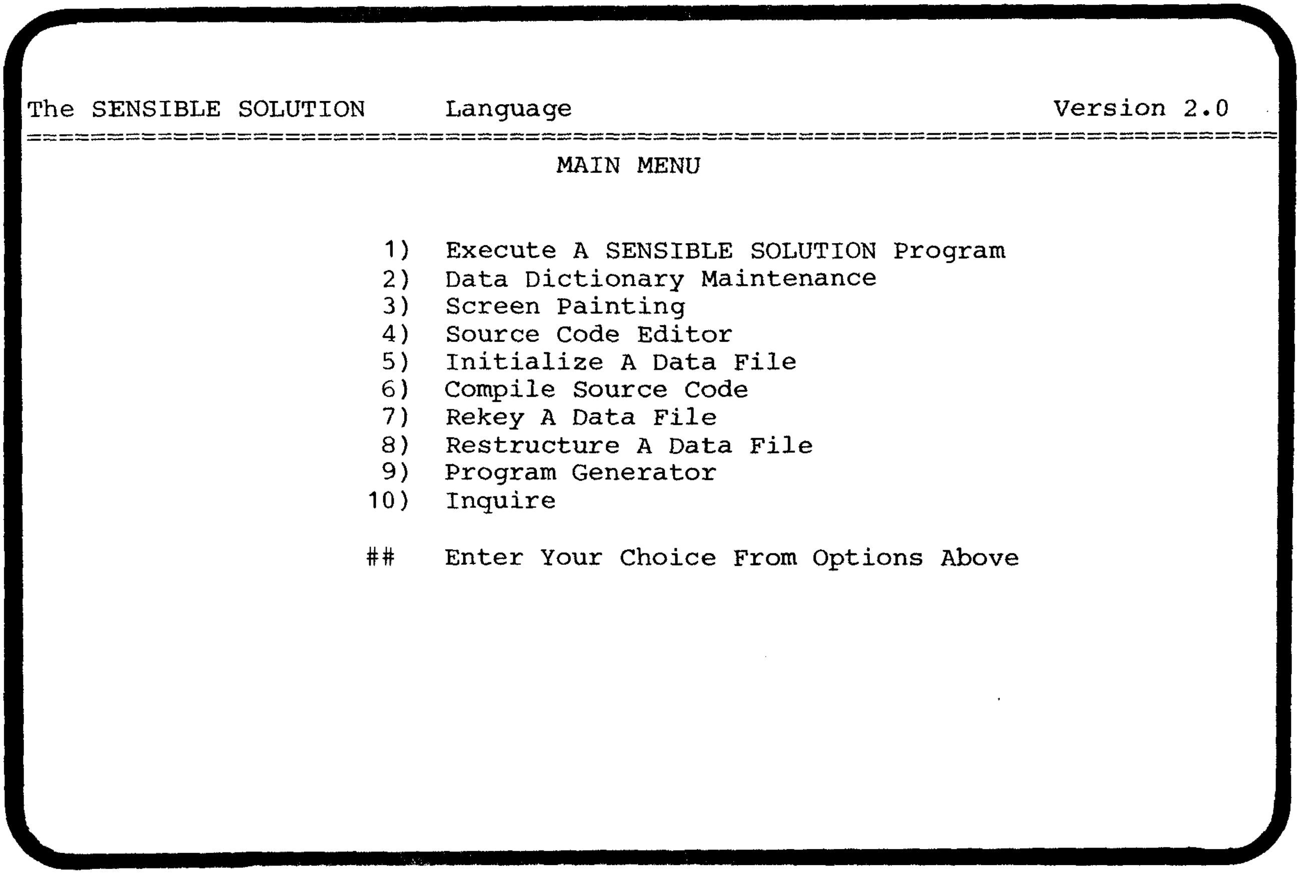
Position your cursor next to the "Work Phone" label and directly below the home phone area code field. Press **[ESC]** and **[RETURN]** to add the work area code field.

Work Phone	Field name:	PHON.AREAWORK	(area code)
	File:	PHONLIST	**press [RETURN]**
	Field type:	N	
	Size:	5	
	Number of decimal places:	0	
	Key:	N	
	Entry mask:	\(###\)	
	Save:	Y	

Position your cursor appropriately to place the work phone number. Press **[ESC] [RETURN]**.

Work Phone	Field name:	PHON.NUMWORK	(work phone number)
	File:	PHONLIST	**press [RETURN]**
	Field type:	A	

After "saving" your work, press **[ESC]** to return to the Main Menu --



```
The SENSIBLE SOLUTION     Language                                    Version 2.0
==============================================================================
                                 MAIN MENU


                      1)   Execute A SENSIBLE SOLUTION Program
                      2)   Data Dictionary Maintenance
                      3)   Screen Painting
                      4)   Source Code Editor
                      5)   Initialize A Data File
                      6)   Compile Source Code
                      7)   Rekey A Data File
                      8)   Restructure A Data File
                      9)   Program Generator
                     10)   Inquire

                     ##    Enter Your Choice From Options Above
```

Size:	8
Key:	N
Entry mask:	###-####
Save:	Y

Position your cursor appropriately to add the work phone extension field. Press **[ESC] [RETURN]**.

Work Phone	Field name:	PHONE.EXTWORK	(work extension)
	File:	PHONLIST	**press [RETURN]**
	Field type:	N	
	Size:	3	
	Number of decimal places:	0	
	Key:	N	
	Save:	Y	

Good job! Our work with the Screen Painter is complete so press **[ESC] [Q]** and then **[RETURN]** to accept the default "yes" to the question

Do you want to save the changes? Yes No

Then press **[ESC]** again to return to the Main Menu.

Now that we have painted a screen and defined the fields within the screen, it's time to create a program. Select "Main Menu 9) Program Generator" and see displayed:

Enter the name of the screen format ?:????????

The first **"?"** followed by **":"** is provided for you to specify the disk drive where your program is located. If everything is running from the same drive, just press your **[SPACE BAR]** to jump past it and type **PHONLIST.**

The SENSIBLE SOLUTION program generator will go through five phases:

- -- Checking fields
- -- Initializing Data files
- -- Generating Command Source file
- -- Checking for Target Labels
- -- Checking for Goto/Gosubs

... and for each phase tells which line of the command file it is scanning. This running comment indicates that the program generator is working. When it finishes you will be returned to the Main Menu.

Briefly, SENSIBLE SOLUTION performed three main steps here: data file

Cursor Screen Controls --

These controls operate when the execution control menu is off. They allow you to move your cursor from field to field, and to edit your data entry.

[UP ARROW] = Move cursor to beginning of previous field. (Locks at topmost field)

[DOWN ARROW] = Move cursor to beginning of next field. (Locks at lowest field)

[RIGHT ARROW] Move cursor one position right within field. Value of the field is unchanged.

[LEFT ARROW] = Move cursor one position left within field. Value of the field is unchanged.

[^D] = Deletes the character "under the cursor". Remaining characters in field field shift left to fill, and a blank appears at right end of field.

[^I] = Inserts a blank "under the cursor". Remaining characters in field shift right. Rightmost character in field is lost.

[^U] = Clears the displayed value from the field window.

initialization, source code generation, and source code compilation. Data file initialization is a process by which SENSIBLE SOLUTION creates data files on the disk. It works by reading the field definitions stored in the Data Dictionary and then creating the file that matches those definitions. The source code necessary to maintain files is automatically generated by SENSIBLE SOLUTION. Compilation is the process of creating pseudo code that the SENSIBLE SOLUTION executive program can run.

We will now take our new SENSIBLE SOLUTION program for a quick spin! Select "Main Menu 1) Execute a SENSIBLE SOLUTION Program."

On your screen you will see:

Enter name of .RUN file to be executed: ?:******

Again, the first **"?"** followed by **":"** is provided for you to specify the disk drive where your program is located. If everything is running from the same drive, just press your **[SPACE BAR]** to jump past it and enter **PHONLIST.** Your PHONLIST screen will be displayed and ready for input.

With the cursor located at the **name** field window, type in a friend's name (last name first, so the indexing key can find the surname). If the name fills the entire field, the cursor automatically jumps to the next field window **Home Phone area code;** otherwise, press **[RETURN]** to tell the computer that the field entry is complete.

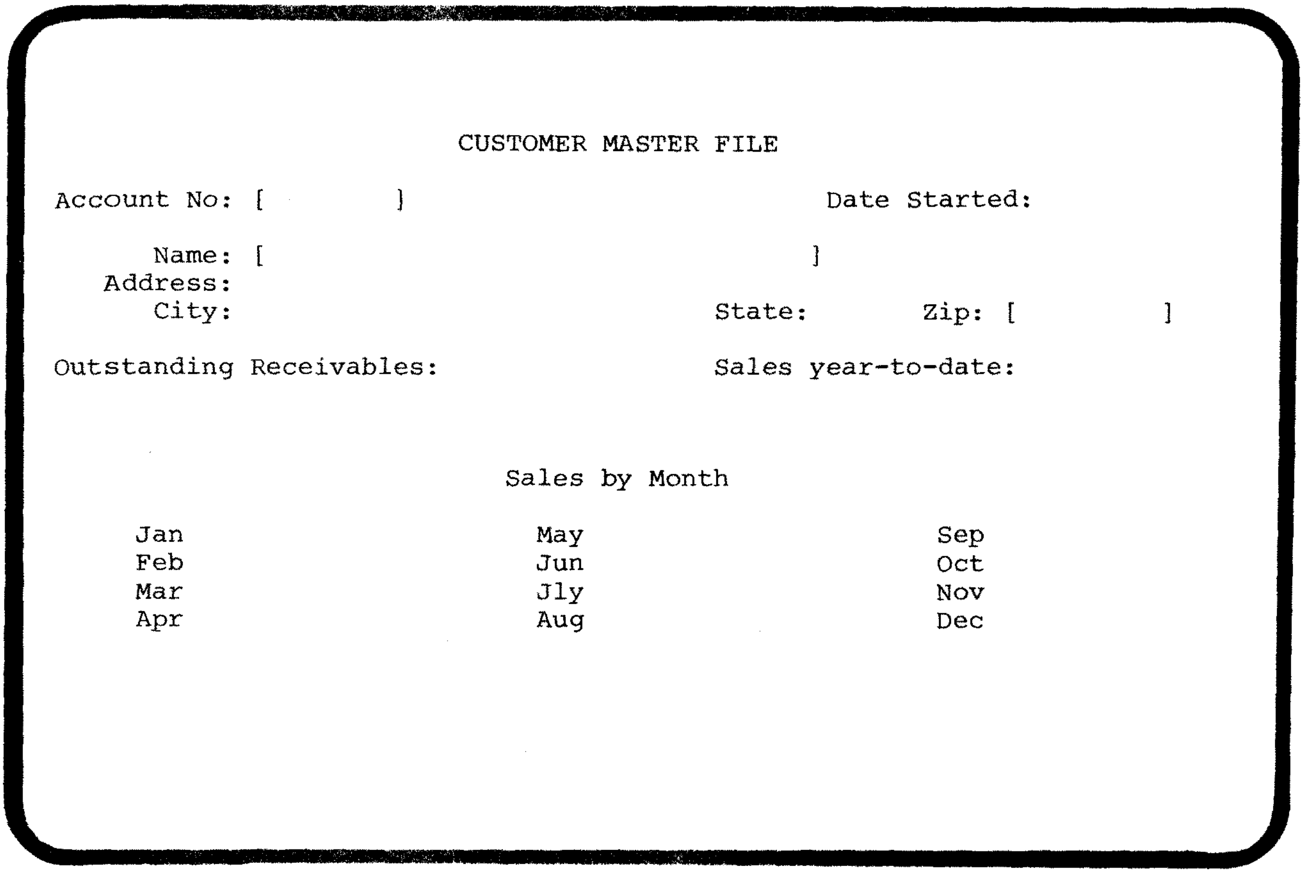
Continue down the screen entering home phone, extension, work phone, and work phone extension. You can **skip** a field (leave it blank) by using the **(NEXT FIELD) [DOWN ARROW]** key or pressing **[RETURN]**. **(PREVIOUS FIELD) [UP ARROW]** will let you go back to correct a previous field.

You may edit the contents of a field with the **(DELETE CHARACTER) [^D]** and **(INSERT CHARACTER) [^I]** keys. **(INSERT CHARACTER) [^I]** bumps everything in the field one space to the right to make room for a new character under the cursor. Characters at the right end of the field fall off and are lost.

You can save the record even if you haven't filled in all the fields. Just press **[ESC] [S] (SAVE record)** and the computer will ask you to confirm **SAVE RECORD? (Y/N).** Take a look at the entry. If it's right, press **[Y]**. The computer will save the new record and insert all the indexing keys in their proper order.

Enter several records and use the menu options to view your records. All done? Then let's **[ESC] [Q]** to the Main Menu. It's time to take the skills you've acquired and create a more sophisticated screen and program which we can edit with the Source Code Editor. Let's begin.

Paint this screen --



```
                         CUSTOMER MASTER FILE

Account No: [        ]                          Date Started:

      Name: [                                  ]
   Address:
      City:                              State:       Zip: [         ]

Outstanding Receivables:                 Sales year-to-date:



                            Sales by Month

     Jan                      May                      Sep
     Feb                      Jun                      Oct
     Mar                      Jly                      Nov
     Apr                      Aug                      Dec
```

LESSON 3
Creating Applications with the SENSIBLE SOLUTION

CUSTFILE is a customer master record program. It summarizes all the information about one particular customer.

Just as before, select "Main Menu 3) Screen Painting." The type of format you want to load is a **Screen format** so press **[RETURN]**. Now press your **[SPACE BAR]** and type **CUSTFILE.**

The question:

```
              +------+
    New file? | Yes  |  No
              +------+
```

will appear on your display terminal.

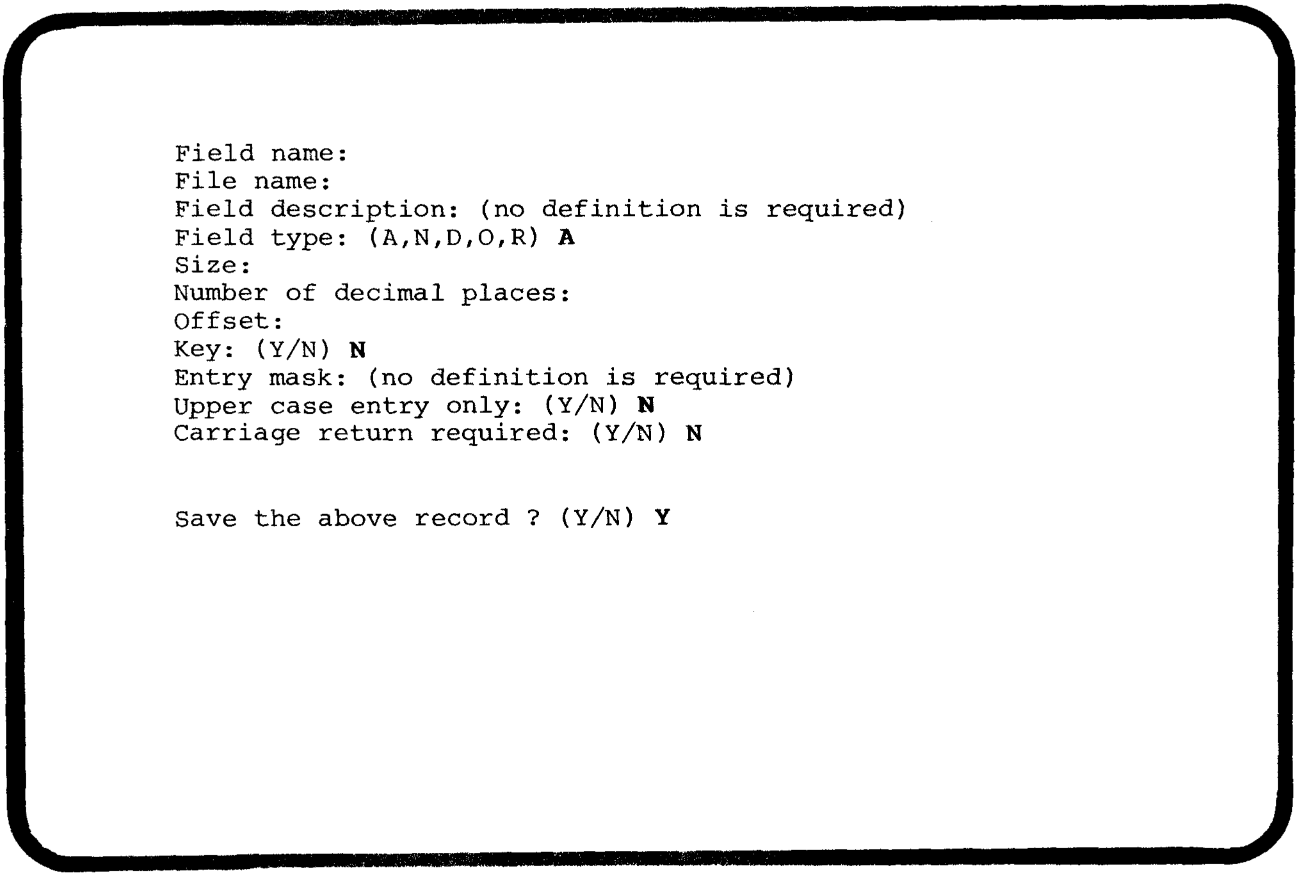
"Yes" is the default answer. Press **[RETURN]** and continue.

Paint the frame as shown on the opposite page. Don't forget to leave room for the field windows!

Now we're going to define and place the field windows, just as we've done before. Position your cursor appropriately, **[ESC]** and "Add" each of the fields as follows:

LABEL	**QUESTION**	**ANSWER**	**NOTES**
Account No.	Field name:	CUS.CUSCODE	
	File:	CUSTFILE	default value--
	Field type:	N	**press [RETURN]**
	Size:	8	
	Number of decimal places:	0	
	Key:	Y	
	Save:	Y	
Date started	Field name:	CUS.DATE	
	File:	CUSTFILE	**press [RETURN]**
	Field type:	D	
	Key:	N	
	Save:	Y	

While using **[ESC] [A] Add fld**, the listed entries represent the default answers obtained by pressing **[RETURN] or [DOWN ARROW]**.



```
Field name:
File name:
Field description: (no definition is required)
Field type: (A,N,D,O,R) A
Size:
Number of decimal places:
Offset:
Key: (Y/N) N
Entry mask: (no definition is required)
Upper case entry only: (Y/N) N
Carriage return required: (Y/N) N


Save the above record ? (Y/N) Y
```

Note: You can use **[UP ARROW] and [DOWN ARROW]** to position your cursor next to any field specification you wish to alter. Press **[^U]** to clear the current definition and re-enter the specification you desire.

Name	Field name	CUS.NAME	
	File:	CUSTFILE	**press [RETURN]**
	Field type:	A	
	Size:	34	
	Key:	Y	
	Upper case entry only:	Y	
	Save:	Y	
Address	Field name:	CUS.ADDR	
	File:	CUSTFILE	**press [RETURN]**
	Field type:	A	
	Size:	20	
	Key:	N	
	Upper case entry only:	Y	
	Save:	Y	
City	Field name:	CUS.CITY	
	File:	CUSTFILE	**press [RETURN]**
	Field type:	A	
	Size:	20	
	Key:	N	
	Upper case entry only:	Y	
	Save:	Y	
State	Field name:	CUS.STATE	
	File:	CUSTFILE	**press [RETURN]**
	Field type:	A	
	Size:	2	
	Key:	N	
	Upper case entry only:	Y	
	Save:	Y	
Zip	Field Name:	CUS.ZIP	
	File:	CUSTFILE	**press [RETURN]**
	Field type:	A	
	Size:	9	
	Key:	Y	
	Save:	Y	
Receivables	Field name:	CUS.RECEIVE	
	File:	CUSTFILE	**press [RETURN]**
	Field type:	N	
	Size:	12	
	Number of decimal places:	2	(dollars & cents)
	Key:	N	
	Save:	Y	

Use **[ESC] [H] (Hard copy)** to obtain a listing like this:

```
@:CUSTFILE.SCC screen format listing        Page No: 0001

                              CUSTOMER MASTER FILE

  Account No: [########]                                Date Started: mm/dd/yy

        Name: [**********************************]
     Address:  ********************
        City:  ********************       State: **    Zip: *********

  Outstanding Receivables: #########.##    Sales year-to-date: #########.##


                                 Sales by Month

       Jan #######.##           May #######.##           Sep #######.##
       Feb #######.##           Jun #######.##           Oct #######.##
       Mar #######.##           Jly #######.##           Nov #######.##
       Apr #######.##           Aug #######.##           Dec #######.##


Field name        File     Size   Col  Row  Key
---------------   -------- ----   ---  ---  ---
CUS.CUSCODE       CUSTFILE  008   016   04   Y
CUS.DATE          CUSTFILE    8   065   04   N
CUS.NAME          CUSTFILE  034   016   06   Y
CUS.ADDR          CUSTFILE  020   016   07   N
CUS.CITY          CUSTFILE  020   016   08   N
CUS.STATE         CUSTFILE  002   051   08   N
CUS.ZIP           CUSTFILE  009   062   08   Y
CUS.RECEIVE       CUSTFILE  012   028   10   N
CUS.SALES         CUSTFILE  012   064   10   N
CUS.MONTH01       CUSTFILE  010   012   16   N
CUS.MONTH05       CUSTFILE  010   037   16   N
CUS.MONTH09       CUSTFILE  010   062   16   N
CUS.MONTH02       CUSTFILE  010   012   17   N
CUS.MONTH06       CUSTFILE  010   037   17   N
CUS.MONTH10       CUSTFILE  010   062   17   N
CUS.MONTH03       CUSTFILE  010   012   18   N
CUS.MONTH07       CUSTFILE  010   037   18   N
CUS.MONTH11       CUSTFILE  010   062   18   N
CUS.MONTH04       CUSTFILE  010   012   19   N
CUS.MONTH08       CUSTFILE  010   037   19   N
CUS.MONTH12       CUSTFILE  010   062   19   N
```

Sales	Field name:	CUS.SALES	
	File	CUSTFILE	**press [RETURN]**
	Field type:	N	
	Size:	12	
	Number of decimal places:	2	
	Key:	N	
	Save:	Y	
Jan	Field name:	CUS.MONTH01	
	File:	CUSTFILE	**press [RETURN]**
	Field type:	N	
	Size:	10	
	Number of decimal places:	2	
	Key:	N	
	Save:	Y	
Feb	Field name:	CUS.MONTH02	
	File:	CUSTFILE	**press [RETURN]**
	Field type:	N	
	Size:	10	
	Number of decimal places:	2	
	Key:	N	
	Save:	Y	

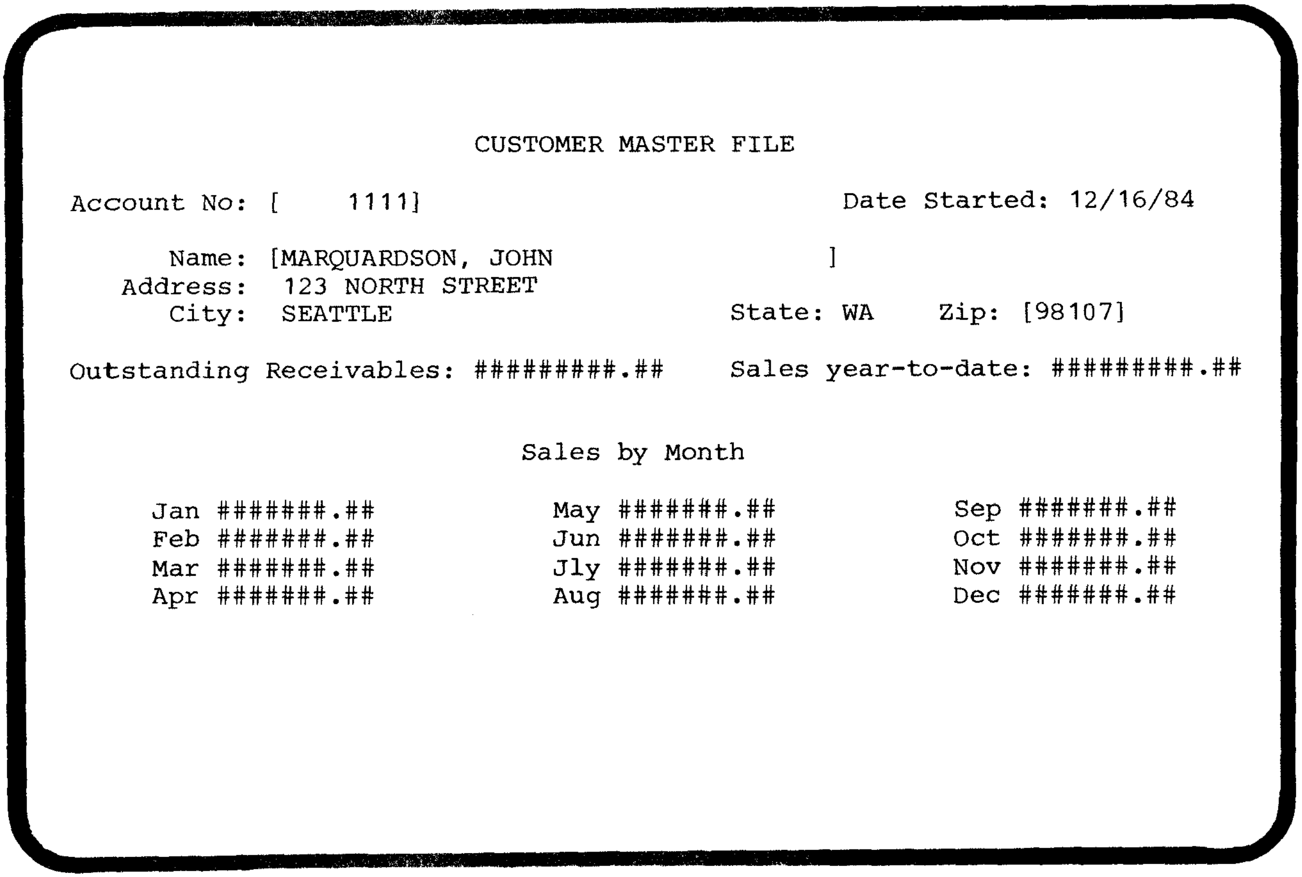
Make entries for the Mar through Dec labels called **CUS.MONTH03, CUS.MONTH04** and so on, but otherwise identical to **CUS.MONTH01.** MONTH01 consists of five letters and two numbers not six letters and one number. Don't confuse zero with the letter "O".

This completes our field definitions for the **CUSTFILE** screen. If you would like a printed copy of your screen as defined, press **[ESC] [H] (Hard copy)** and the listing shown on the facing page will be printed on your system printer. You may also examine a particular field by positioning the cursor at the beginning of the field window then pressing **[ESC] [S] (Show fld).** The field definition will appear at the top of the display.

You can change the definition of a field within **Screen Painting** by using **(Field chg) [ESC] [F]** and redefining any field specification. Pressing **[ESC] [R] (Remove fld)** keys will take a field off the screen but does not remove it from the **Data Dictionary.** If you remove a field and use the same field name again, the existing definition will be used automatically.

Executing the CUSTFILE program--

In the name field, enter last name first.



```
                         CUSTOMER MASTER FILE

Account No: [    1111]                       Date Started: 12/16/84

      Name: [MARQUARDSON, JOHN                   ]
   Address:  123 NORTH STREET
      City:  SEATTLE                     State: WA    Zip: [98107]

Outstanding Receivables: #########.##    Sales year-to-date: #########.##

                            Sales by Month

     Jan #######.##            May #######.##            Sep #######.##
     Feb #######.##            Jun #######.##            Oct #######.##
     Mar #######.##            Jly #######.##            Nov #######.##
     Apr #######.##            Aug #######.##            Dec #######.##
```

Now that the **CUSTFILE** screen has been defined, press **[ESC] [Q]**. The computer will ask

Do you want to save the changes? Y/N

Press **Y.** The screen definition will be saved and you will be asked once more what type of format you wish to load into the Screen Painter. If you had more screens to create, you would simply name another and begin to paint and define it. At this time we are finished screen painting, so press **[ESC]** to return to the Main Menu.

Now let's use "Main Menu 9) Program Generator" and create a program using our CUSTFILE screen format.

The Program Generator will prompt

Enter the name of the screen format: ?:******

Type **[SPACE BAR] CUSTFILE.** The program generator will report on its progress as it works. When the program generation is complete, the SENSIBLE SOLUTION main menu will reappear. Now we're ready to execute the program.

We will give this new program a trial run; then later, we will modify it to connect with our sales-order and payment-entry screens.

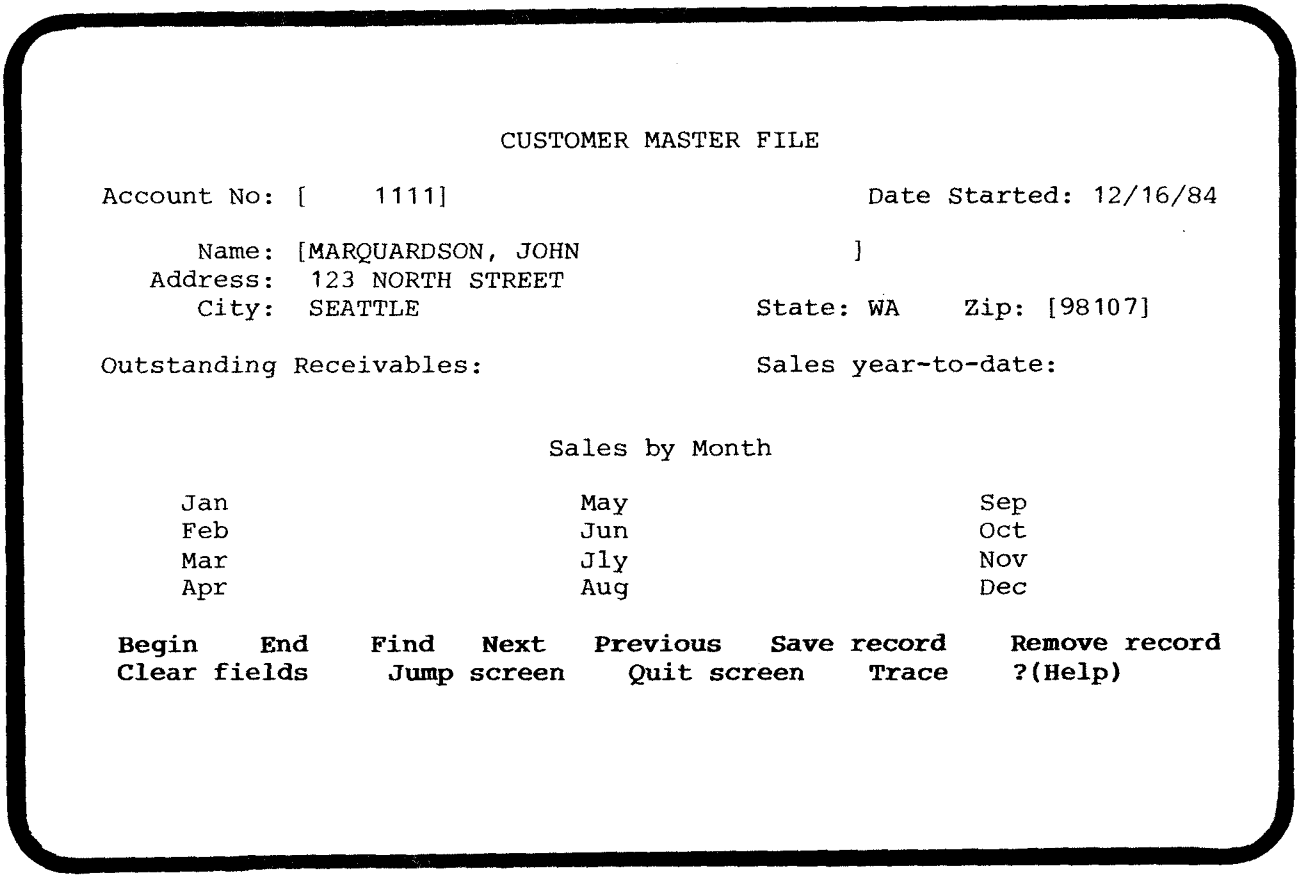
Select "Main Menu 1) Execute A Sensible Solution Program." The display will clear and ask you to **Enter name of .RUN file to be executed.** Type **CUSTFILE** (you don't have to press **[RETURN]**). The screen we just defined will appear on the terminal display and allow you to enter, examine, or update information in the Customer Master File.

The same ESC Menu and command keys that you used with the MAILLIST and PHONLIST programs are available to you to aid in entering data to the screen. Please enter records for account numbers **1111, 2222, 3333, 4444,** and so on. Specify account number, date started, name, address, city, state, and zip code only. In the name field, enter last name first. Use your own customers' names and addresses for data, or make up a few imaginary customers. We'll need five or six customer records for later when we create and use the **sales-order** and **payment-entry** screens.

Again, play with the entries and get the feel of data entry and the control keys. A few things to try:

-- The cursor automatically moves to the next field when you fill the window of the current field. A **[RETURN]** and a **(NEXT FIELD) [DOWN**

Use the execution control menu to experiment with your CUSTFILE program --



```
                              CUSTOMER MASTER FILE

   Account No: [     1111]                         Date Started: 12/16/84

         Name: [MARQUARDSON, JOHN                    ]
      Address:  123 NORTH STREET
         City:  SEATTLE                        State: WA    Zip: [98107]

   Outstanding Receivables:                    Sales year-to-date:


                                 Sales by Month

        Jan                        May                        Sep
        Feb                        Jun                        Oct
        Mar                        Jly                        Nov
        Apr                        Aug                        Dec

    Begin    End     Find   Next   Previous   Save record   Remove record
    Clear fields      Jump screen    Quit screen    Trace    ?(Help)
```

ARROW] will move you to the next field window. **(PREVIOUS FIELD) [UP ARROW]** moves you back one "window".

-- **(CLEAR FIELD) [^U]** erases the data in the field where your cursor is located. **[ESC] [C] (CLEAR SCREEN)** erases the entire entry and repositions the cursor in the first field at the top of the screen. Remember, you must clear the screen before entering a new record. Changingthe field values on an existing record will overwrite the old data and replace it with the new.

-- **(DELETE CHARACTER)** [^D] and **(INSERT CHARACTER)** [^I] along with **(CURSOR LEFT) [LEFT ARROW]** and **(CURSOR RIGHT) [RIGHT ARROW]** are useful in correcting typing mistakes. They work on both old data and new field entries.

-- Searches **[ESC] [FIND], [ESC] [B] (find BEGINning record), [ESC] [E] (find ENDing record)** will fail if your cursor is not in an indexing-key field. **[ESC] [F]** begins its scan with all the data shown in the field, even if part of the data was already there before you began typing. Remember to **[CLEAR FIELD] (^U)** before searching.

-- **[ESC] [N] (find NEXT record)** and **[ESC] [P] (find PREVIOUS record)** need to know what the "current index-key" is. They fail if a **[ESC] [F]**, **[ESC] [B]**, or **[ESC] [E]** has not been done first.

-- **[ESC] [S] (SAVE record)** and **[ESC] [R] (REMOVE record)** will both work from anywhere on a screen, even if some of the fields have not been typed in. Of course, removing a new record will fail because it only exists "on the display" and has not yet been saved to the disk file.

Don't forget to save each individual record. When you have finished entering and examining five or six customer records press **[ESC] [Q] (QUIT)**, and you will return to the main menu.

Now that you have painted a screen and entered data into fields created by SENSIBLE SOLUTION, let's look at our program and see how we can enhance it.

```
@:CUSTFILE.SRR source file listing          Page No: 0001

           remark
           trap SAVE goto SAVE.GRP
           trap DELETE goto DELT.GRP
           mount screen CUSTFILE
START      enter CUS.CUSCODE
           enter CUS.DATE
           enter CUS.NAME
           enter CUS.ADDR
           enter CUS.CITY
           enter CUS.STATE
           enter CUS.ZIP
           enter CUS.RECEIVE
           enter CUS.SALES
           enter CUS.MONTH01
           enter CUS.MONTH05
           enter CUS.MONTH09
           enter CUS.MONTH02
           enter CUS.MONTH06
           enter CUS.MONTH10
           enter CUS.MONTH03
           enter CUS.MONTH07
           enter CUS.MONTH11
           enter CUS.MONTH04
           enter CUS.MONTH08
           enter CUS.MONTH12
SAVE.GRP   save rec in file CUSTFILE confirm / clear buffer
           goto START
DELT.GRP   delete rec in file CUSTFILE confirm
           goto START
```

LESSON 4
Editing a SENSIBLE SOLUTION Program

Select "Main Menu 4) Source Code Editor." The screen will then display:

```
 +--------------------------------------+
 | Load command-file source for editing |   Quit
 +--------------------------------------+

The SENSIBLE SOLUTION Command Source Editor  V2.0C
==========================================================================
```

The cursor will be positioned over "Load command-file source for editing". You want to load a program (CUSTFILE) so press **[RETURN]**.

The computer will ask:

Enter name of the source file: ?:????????

The first "?" followed by ":" is provided for you to specify the drive location of your **.SRR** file (the CUSTFILE source code file). In this case CUSTFILE.SRR is located on the same drive as the one you are presently working on, so press your **[SPACE BAR]** and enter the name of your source file **CUSTFILE.** There is no need to press **[RETURN].** The Source Code Editor accepts file names up to eight characters long. When you enter an eight character file name the Editor automatically loads the specified file. You must press **[RETURN]** only if your file name does not fill the space provided (if it is shorter than eight characters in length). If you already did press **[RETURN]** an error message will be displayed--press **[RETURN]** again and you're ready to go.

The computer will display the CUSTFILE source code file on your screen. If the file name was misspelled or you are opening a new file the computer will ask:

New File? Yes No

If you wished to open a new file you would respond with **[Y]**. The computer then would give you a blank space to write source code. If you don't want to open a new file (maybe you misspelled the file name) respond with **[N]**. This will bring you back to "Load" and you can try again.

@:CUSTFILE.SRR source file listing   Page No: 0001

```
           remark
           trap SAVE goto SAVE.GRP
           trap DELETE goto DELT.GRP
           mount screen CUSTFILE
START      enter CUS.CUSCODE
           enter CUS.DATE
           enter CUS.NAME
           enter CUS.ADDR
           enter CUS.CITY
           enter CUS.STATE
           enter CUS.ZIP
           enter CUS.RECEIVE
           enter CUS.SALES
           enter CUS.MONTH01
           enter CUS.MONTH05
           enter CUS.MONTH09
           enter CUS.MONTH02
           enter CUS.MONTH06
           enter CUS.MONTH10
           enter CUS.MONTH03
           enter CUS.MONTH07
           enter CUS.MONTH11
           enter CUS.MONTH04
           enter CUS.MONTH08
           enter CUS.MONTH12
SAVE.GRP   save rec in file CUSTFILE confirm / clear buffer
           goto START
DELT.GRP   delete rec in file CUSTFILE confirm
           goto START
```

The computer will locate **CUSTFILE.SRR** and bring it into memory for examination and editing. You will notice that the computer generated source code file reads very much like English.

The Source Code Editor offers 15 different menu options for entering or modifying a program. The editing options are displayed on a menu like this:

```
  Change line    Insert line    Delete line    Begin source    End source
  Previous page    Next page    Read block    Mark block    Write block
  Transfer block    delete blocK    Hard Copy    Find line    Quit
@:CUSTFILE.SRR       On line: 0001    Tot lines: 0001  Insert off
==============================================================================
```

Let's look at each menu selection:

[Q] = Quit the screen you are currently viewing and return to the previous screen used by the system. You have the option to leave the Source Code editor completely, or to leave the current file and begin edit work on a new program.

[P] = Using this selection you may scroll the source code down 8 lines. On the screen you will see the top line move to the middle of the screen, revealing the 8 lines that came before it. A right angle bracket character ">" will mark the new position of this line.

[N] = Using this selection you may scroll the source code up 8 lines. On the screen you will see the bottom line move to the middle of the screen, revealing the 8 lines that follow it. A right angle bracket character ">" will mark the new position of this line.

[B] = This selection finds the beginning command in the source file. It sets the line prompt ">" at the first line of code.

[E] = This selection finds the ending command in the source file. It sets the line prompt ">" at the last line of code.

[I] = This selection will insert a command line following the command marked by the line prompt ">". The **COMMANDS** menu is displayed if you select insert mode. Note that the insert will <u>follow</u> the line being pointed at, not precede it.

You may obtain a print out like this one by using the Source Code Editor menu selection **[ESC] [H] (Hard copy).**

```
@:CUSTFILE.SRR source file listing          Page No: 0001

0001                remark
0002                trap SAVE goto SAVE.GRP
0003                trap DELETE goto DELT.GRP
0004                mount screen CUSTFILE
0005 START          enter CUS.CUSCODE
0006                enter CUS.DATE
0007                enter CUS.NAME
0008                enter CUS.ADDR
0009                enter CUS.CITY
0010                enter CUS.STATE
0011                enter CUS.ZIP
0012                enter CUS.RECEIVE
0013                enter CUS.SALES
0014                enter CUS.MONTH01
0015                enter CUS.MONTH05
0016                enter CUS.MONTH09
0017                enter CUS.MONTH02
0018                enter CUS.MONTH06
0019                enter CUS.MONTH10
0020                enter CUS.MONTH03
0021                enter CUS.MONTH07
0022                enter CUS.MONTH11
0023                enter CUS.MONTH04
0024                enter CUS.MONTH08
0025                enter CUS.MONTH12
0026 SAVE.GRP       save rec in file CUSTFILE confirm / clear buffer
0027                goto START
0028 DELT.GRP       delete rec in file CUSTFILE confirm
0029                goto START
```

[D] = This selection allows you to delete the line at the position marked by the line prompt ">".

[C] = This selection will change the command at the line marked by the line prompt ">". The **COMMANDS** menu is displayed if you select change mode.

[M] = Using this selection you may mark the beginning or the end of a list of command lines in your program. Having thus defined a block of text, you may perform file/block editing operations (e.g., deleting a block, transferring a block to a new position in the program).

[W] = A marked block of command lines in your program is written to a disk file. This action will not clear your block marks.

[R] = A block of command lines which you have sent to a disk holding file will be "read" (inserted) into the program you are editing. The inserted block will follow the line marked by the line prompt ">".

[T] = Use this selection to transfer a marked block of command lines to a new position within your program. The transferred block will follow the line marked by the line prompt ">". After use, the block marks are deleted and would have to be reset in order to move the block again.

[K] = Use this selection to delete a marked block of command lines from your program.

[H] = Use this selection to print a listing of the source code file (.SRR).

[F] = This selection allows you to search your source file for a specific label, field name, or line number. The search for a line proceeds only from the position in the program at which you issue this command. You can not search backwards through a program. The last criteria that you specified is stored, allowing you to repeat your search rapidly.

Line numbers do not display on the terminal; they only appear on hard copy --

CUSTFILE.SRR source file listing:

```
LINE #   LABEL            COMMAND

  0001                    remark
  0002                    trap SAVE goto SAVE.GRP
  0003                    trap DELETE goto DELT.GRP
  0004                    mount screen CUSTFILE
  0005   START            enter CUS.CUSCODE
  0006                    enter CUS.DATE
  0007                    enter CUS.NAME
  0008                    enter CUS.ADDR
  0009                    enter CUS.CITY
  0010                    enter CUS.STATE
  0011                    enter CUS.ZIP
  0012                    enter CUS.RECEIVE
  0013                    enter CUS.SALES
  0014                    enter CUS.MONTH01
  0015                    enter CUS.MONTH05
  0016                    enter CUS.MONTH09
  0017                    enter CUS.MONTH02
  0018                    enter CUS.MONTH06
  0019                    enter CUS.MONTH10
  0020                    enter CUS.MONTH03
  0021                    enter CUS.MONTH07
  0022                    enter CUS.MONTH11
  0023                    enter CUS.MONTH04
  0024                    enter CUS.MONTH08
  0025                    enter CUS.MONTH12
  0026   SAVE.GRP         save rec in file CUSTFILE confirm / clear buffer
  0027                    goto START
  0028   DELT.GRP         delete rec in file CUSTFILE confirm
  0029                    goto START
```

When you select Insert **[ESC] [I]** or Change **[ESC] [C]** mode by pressing the appropriate keys, and then pressing **[RETURN]**, the following **COMMAND** options will be displayed:

Enter = **I**f **G**o **M**ount **S**ave rec **D**elete rec **C**lear buffer
Find **P**rint **T**rap **E**xecute **!** remark **L**ock **U**nlock

Each of the options (activated by pressing the bold print Capital character, or highlighting your choice and pressing **[RETURN]** is a command. Each command has a subset of functions. You will be asked questions about each of these functions. Your answer will determine the specifics of how the command is to perform during program execution.

The SENSIBLE SOLUTION is quite different from a language like BASIC, and the source listing is different, too. Let's look at it piece by piece.

The **line number** is just an indicator to let you tell the Source Code Editor which command line you want to work on. Line numbers are not referenced within the program itself. The Source Code Editor provides a status line telling you the name of the .SRR file that is "opened", the line you can work with -- "On line: ####", and the total number of lines in the program -- "Tot lines: ####". On the terminal, line numbers do not display next to each command line. On hard copy they will be printed out. (Use **[ESC] [H]** (Hard copy) to obtain such a listing.)

The **label** indicates which line will be the "target" of a control transfer. GOTO and GOSUB, for instance, are commands that transfer the control of the program to a specified label. For example:

trap DELETE goto DELT.GRP

SENSIBLE SOLUTION programs consist of collections of **commands.** Each command will require one or more parameters.

The source file listing is used to examine the flow of logic within a program. You do not edit a text file that looks like a SENSIBLE SOLUTION listing and then compile that text into an executable program. Instead, using the **Source Code Editor** you specify the line you want to create and the editor will prompt you for the command and parameters automatically.

This is an essential and important difference between the SENSIBLE SOLUTION and language's like BASIC or COBOL. Because the editor prompts you for the command and all necessary parameters, syntax errors can never happen. The editor will reject a reference to a field that hasn't been defined. The SENSIBLE SOLUTION executive controls all details of data representation.

You can scroll through the program using the **[ARROW]** keys, **Find line, Beg, End, Previous page, and Next page.**

```
@:CUSTFILE.SRR source file listing          Page No: 0001

0001               remark
0002               trap SAVE goto SAVE.GRP
0003               trap DELETE goto DELT.GRP
0004               mount screen CUSTFILE
0005 START         enter CUS.CUSCODE
0006               enter CUS.DATE
0007               enter CUS.NAME
0008               enter CUS.ADDR
0009               enter CUS.CITY
0010               enter CUS.STATE
0011               enter CUS.ZIP
0012               enter CUS.RECEIVE
0013               enter CUS.SALES
0014               enter CUS.MONTH01
0015               enter CUS.MONTH05
0016               enter CUS.MONTH09
0017               enter CUS.MONTH02
0018               enter CUS.MONTH06
0019               enter CUS.MONTH10
0020               enter CUS.MONTH03
0021               enter CUS.MONTH07
0022               enter CUS.MONTH11
0023               enter CUS.MONTH04
0024               enter CUS.MONTH08
0025               enter CUS.MONTH12
0026 SAVE.GRP      save rec in file CUSTFILE confirm / clear buffer
0027               goto START
0028 DELT.GRP      delete rec in file CUSTFILE confirm
0029               goto START
```

By letting the computer deal with these details, SENSIBLE SOLUTION eliminates 90% of the errors that make programs so hard to debug, while leaving the freedom to design program logic in the hands of the programmer. Designers using traditional languages have been suspicious of this approach until they try it. Once a complex program performs flawlessly on the first attempt, they never want to go back!

SENSIBLE SOLUTION programs consist of a number of lines containing a specific SENSIBLE SOLUTION command and appropriate parameters. Programs can be up to 2000 lines long! Most of the programs though, are quite short. For instance, the automatically-generated program **CUSTFILE** is only 29 lines long.

Let's go through the **CUSTFILE** listing step-by-step and examine what it does. Remember, this entire program -- labels, commands and parameters -- was automatically created when we "painted" our screen using "Main Menu 3) Screen Painting" and then generated a program using "Main Menu 9) Program Generator."

You can scroll through your CUSTFILE program very easily. The **[UP ARROW]** and **[DOWN ARROW]** keys will move you through your program one line at a time in either direction. The **Find line** option allows you to look for a specific command, label, or line number. **Beg** and **End** move the line prompt "**>**" to the very beginning or ending of the command file.

Line 2 **trap SAVE goto SAVE.GRP**

The concept of setting up a **Trap** is a very important one in the SENSIBLE SOLUTION. Nothing happens immediately when a **Trap** is set up. Instead, the command tells the computer where to transfer control when a particular condition or event occurs during the execution of the program.

In this example, whenever SENSIBLE SOLUTION senses that the program operator has pressed the **(Save record) [ESC] [S]** keys it will exit from the command it's executing, usually an **Enter** command, and search for the specified line label. Once it finds that label, it continues program execution from that point. In this case, program control is diverted to **SAVE.GRP** (a set of commands that saves the current record, clears the screen and transfers control to **START** to allow a new record entry).

Line 3 **trap DELETE goto DELT.GRP**

This command is executed whenever the **(Remove record) [ESC] [R]** option is entered at the terminal by the program operator. It transfers

You can scroll through the program using the **[ARROW]** keys, **Find line, Beg, End, Previous page, and Next page.**

```
@:CUSTFILE.SRR source file listing          Page No: 0001

0001                remark
0002                trap SAVE goto SAVE.GRP
0003                trap DELETE goto DELT.GRP
0004                mount screen CUSTFILE
0005 START          enter CUS.CUSCODE
0006                enter CUS.DATE
0007                enter CUS.NAME
0008                enter CUS.ADDR
0009                enter CUS.CITY
0010                enter CUS.STATE
0011                enter CUS.ZIP
0012                enter CUS.RECEIVE
0013                enter CUS.SALES
0014                enter CUS.MONTH01
0015                enter CUS.MONTH05
0016                enter CUS.MONTH09
0017                enter CUS.MONTH02
0018                enter CUS.MONTH06
0019                enter CUS.MONTH10
0020                enter CUS.MONTH03
0021                enter CUS.MONTH07
0022                enter CUS.MONTH11
0023                enter CUS.MONTH04
0024                enter CUS.MONTH08
0025                enter CUS.MONTH12
0026 SAVE.GRP       save rec in file CUSTFILE confirm / clear buffer
0027                goto START
0028 DELT.GRP       delete rec in file CUSTFILE confirm
0029                goto START
```

program control to the command line labeled **DELT.GRP.**

Line 4   mount screen CUSTFILE

This command displays the screen CUSTFILE at the terminal, clears all the field windows to blanks or zeroes, and prepares the computer to enter or display data.

Line 5 START enter CUS.CUSCODE

Beginning at the label **START**, we reach a series of **Enter** commands, each allowing entry to, or editing of a particular field. **Enter** will display the value of the field within the active record. If there is no active record, **Enter** will fill the field with blanks or a zero. When an **Enter** command is executed, the flow of program control "drops through" to the next command unless a control key is used. If you press **(PREVIOUS FIELD) [UP ARROW]** SENSIBLE SOLUTION exits the **Enter** command you are in and moves you to the preceding **Enter** command.

Line 6-25   enter FIELDNAME

Allows the user to enter the necessary information as described above.

Finally, there are two groups of commands that save the record and clear the screen for a new entry **(Save Rec)** or delete an existing record **(Delete rec).** Notice that the "delete" group can **only** be reached by triggering the **Trap** on **Delete** with an **[ESC] [R] (Remove record).**

Line 26 SAVE.GRP save rec in file CUSTFILE confirm / clear buffer

This command saves the current record to memory and clears the buffer if Y is answered to the Save question. Otherwise, the program continues with the next line.

Line 27   goto START

This command sends program control to the command line labeled START.

Line 28 DELT.GRP delete rec in file CUSTFILE confirm

This command deletes the record from memory if Y is answered to the Remove question. Otherwise, program control falls through to the next line.

CUSTFILE.SRR after deleting lines 12 through 25 --

```
CUSTFILE.SRR source file listing          Page No: 0001

0001                remark
0002                trap SAVE goto SAVE.GRP
0003                trap DELETE goto DELT.GRP
0004                mount screen CUSTFILE
0005 START          enter CUS.CUSCODE
0006                enter CUS.DATE
0007                enter CUS.NAME
0008                enter CUS.ADDR
0009                enter CUS.CITY
0010                enter CUS.STATE
0011                enter CUS.ZIP
0012 SAVE.GRP       save rec in file CUSTFILE confirm / clear buffer
0013                goto START
0014 DELT.GRP       delete rec in file CUSTFILE confirm
0015                goto START
```

Line 29 **goto START**

This command sends program control to the command line labeled START.

Some final comments on the program: **START, SAVE.GRP** and **DELT.GRP** are labels that the Program Generator creates automatically. For simple applications, you'll generally find it easiest to create programs directly from screens. You can then edit or combine the resulting programs to enhance their operation.

You need to make some modifications to CUSTFILE.SRR.

Remember, you can scroll through your CUSTFILE program using **[UP ARROW]** and **[DOWN ARROW]** keys. The **Find line** option allows you to look for a specific command, label, or line number. **Beg and End** move the cursor to the very beginning or ending of the command file.

The first thing that we wish to change in CUSTFILE.SRR is lines 12 through 25. We want to create two separate programs (CUSTPYMT and CUSTORD) to gather this information and relate it to the customer master file (CUSTFILE). We don't want to be able to enter this data in the customer master file. Therefore, delete lines CUS.RECEIVE through CUS.MONTH12. Of course you may delete each line, one at a time, using **[D] (Delete line),** but try using the **[M] (Mark block) and [K] (delete blocK)** function to remove lines 12 through 25. These functions will speed up your work considerably.

We want the program to report any duplicates found in the data and display a message on the screen. To do this we must insert a command to check to see if the data already exists. Go into the **insert** mode by pressing **[I]** and notice on the status line that it now display "insert on".

There is a line of 15 *'s displayed in the upper left hand corner of the screen. This field window would be used to enter a label if you wished to label this command. Press **[RETURN]** and the menu of command options is displayed.

Position your line prompt "**>**" at line 5 and insert this command:

if duplicate key using field CUS.CUSCODE goto START

Follow this procedure: Press **[I]** for **IF.**
Press **[D]** for **Duplicate key check**
Type **CUS.CUSCODE** and press **[RETURN].**
Type **START** and press **[RETURN].**
Press **[RETURN]** to save the new line.

The completely modified CUSTFILE program --

```
CUSTFILE.SRR source file listing          Page No: 0001

                remark
                trap SAVE goto SAVE.GRP
                trap DELETE goto DELT.GRP
                mount screen CUSTFILE
     START      enter CUS.CUSCODE
                if duplicate key using field CUS.CUSCODE goto START
                enter CUS.DATE
                enter CUS.NAME
                enter CUS.ADDR
                enter CUS.CITY
                enter CUS.STATE
                enter CUS.ZIP
     SAVE.GRP   save rec in file CUSTFILE confirm / clear buffer
                goto START
     DELT.GRP   delete rec in file CUSTFILE confirm
                goto START
```

This is the only command we wish to insert so **[ESC]** to the Source Code Editor menu.

We are through modifying CUSTFILE.SRR, so let's return to the Main Menu. Press **[Q]** to quit and respond "Yes" to exit completely.

As we made changes in our program using the Source Code Editor, each change in the source code file **(.SRR)** was updated on the disk as it was made. However, unless we **re-compile** our CUSTFILE.RUN file, the new commands we have created will not be executed when we run the CUSTFILE program.

Select "Main Menu 6) Compile Source Code."

On your screen see:

```
           Enter the name of the file to be compiled: ?:********
```

Press your **[SPACE BAR]** and type **CUSTFILE.**

Your screen will then display:

```
Checking for target labels
Checking for Goto/Gosubs
Checking for Screen/Reporter formats
Writing out compiled command file

                        Checking Line ???
```

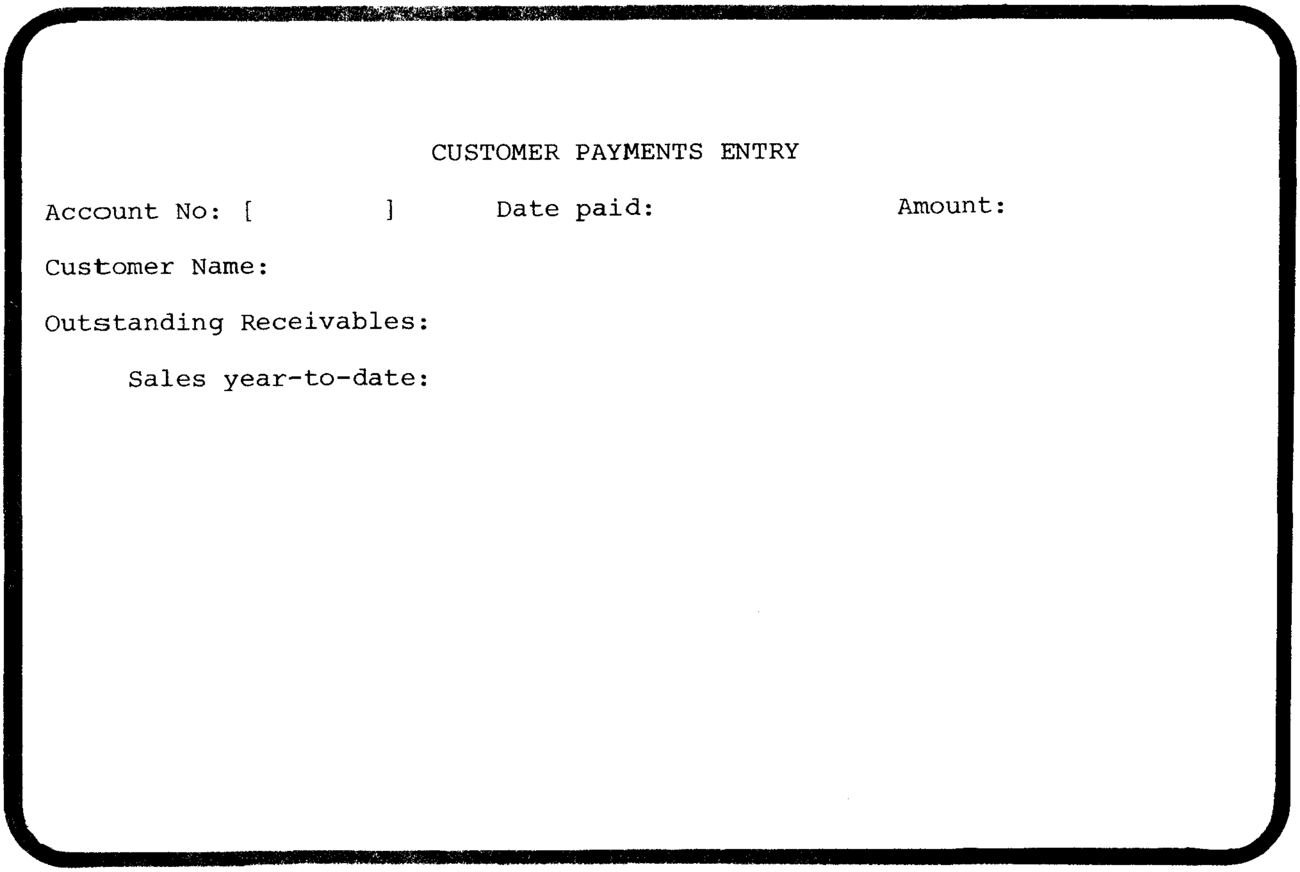
When the compilation is complete, you will be returned to the Main Menu.

The SENSIBLE SOLUTION Language Compiler reads the source code file (.SRR file) and translates it into a code which the computer can read to execute a program. The compiler translates the source code file and creates a command file (.RUN file) which will always be used to execute the CUSTFILE program.

Now that the revised CUSTFILE program is compiled let's go back and execute it! Select "Main Menu 1) Execute a SENSIBLE SOLUTION" program and specify CUSTFILE as the .RUN file to execute. You will discover that the new program obstinately refuses to allow entry of a duplicate account number; nor may you enter data into the monthly sales fields; however, it will allow you to search for an existing account number.

Now we're ready to create the companion sales-order and payment-entry programs.

Paint this CUSTPYMT screen --



```
                         CUSTOMER PAYMENTS ENTRY
Account No: [        ]      Date paid:                Amount:

Customer Name:

Outstanding Receivables:

     Sales year-to-date:
```

LESSON 5
Creating The Customer Payment Program

We'll design the **customer payment** entry screen as the second part of our system. It is used to record receipt of payment on account by customers who owe you money.

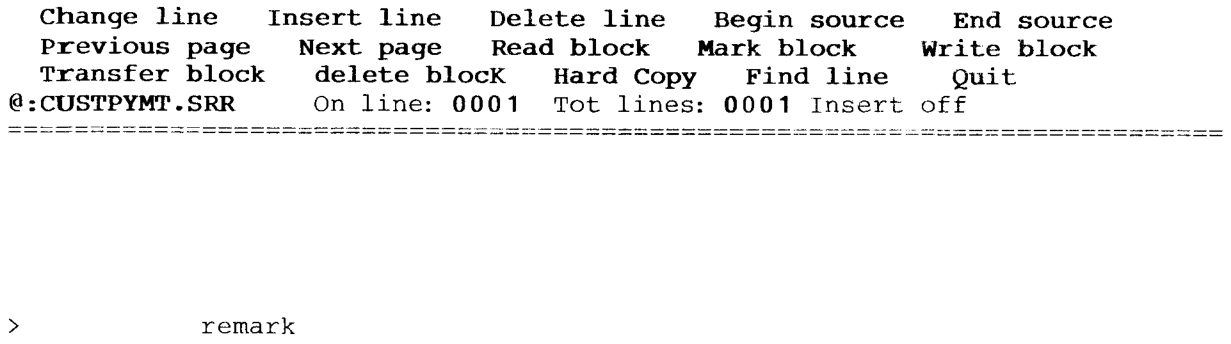
You know how to use "Main Menu 3) Screen Painting" to define a screen. Let's begin by creating the **(.SCC)** screen file name, CUSTPYMT, and painting a screen that looks like the one shown on the opposite page.

Now enter the following field parameters in the appropriate positions on the screen. (Use **[ESC] [A] Add fld.**)

Label	**Computer asks:**	**You answer:**
Account no:	Field name:	PYMT.CUSCODE
	File name:	CUSTPYMT
	Field type:	N
	Size:	8
	Number of decimal places:	0
	Key:	Y
Date paid:	Field name:	PYMT.DATE
	File name:	(CUSTPYMT)
	Field type:	D
	Key:	N
Amount:	Field name:	PYMT.PAYMENT
	File name:	(CUSTPYMT)
	Field type:	N
	Size:	10
	Number of decimal places:	2
	Key:	N
Name:	Name:	CUS.NAME
Receivables:	Name:	CUS.RECEIVE
Sales:	Name:	CUS.SALES

Notice that the SENSIBLE SOLUTION automatically used the already-defined types and formats for **CUS.NAME, CUS.RECEIVE and CUS.SALES.**

Using the Source Code Editor --



```
  Change line   Insert line   Delete line   Begin source   End source
  Previous page   Next page   Read block   Mark block    Write block
  Transfer block   delete blocK   Hard Copy   Find line    Quit
@:CUSTPYMT.SRR     On line: 0001  Tot lines: 0001 Insert off
==============================================================================




>            remark
```

When your screen is correct, **[ESC] (Q)**, and save the screen. The SENSIBLE SOLUTION will save the screen definition. **[ESC]** to the Main Menu.

So far, we have painted a screen and defined files and fields using the Screen Painting Editor. Our file and field definitions are stored in the **Data Dictionary.** Now the data files must be "initialized." We will use "Main Menu 5) Initialize a Data File."

Press **[5]**. The following display wil appear:

Enter the file name: ******

Type **CUSTPYMT.**

The drive location and file name will be displayed in the upper right hand corner of the screen as the program runs. Two files CUSTPYMT.**MS** and CUSTPYMT.**KS** are created at this time. The .MS data file will be used to store all data records entered to the file CUSTPYMT. The .KS file will store keys which will be used by SENSIBLE SOLUTION to locate the desired record in the .MS file.

When the CUSTPYMT file has been initialized, you will again see:

Enter the file name: *****

At this time press **[ESC]** and return to the Main Menu.

We are going to create a customer payment program to write and store a record of each customer payment received. Our payment records will be keyed by the customer code (Account No.). Each time we write a new payment record, we want the program to look up the master customer record by the same code, and fetch the current information for the customer to display on screen.

We will create the CUSTPYMT.SRR file using the Source Code Editor. Considering the changes that you had to make in CUSTFILE, you can see why it is common practice to create all source code with the Editor instead of relying Screen Painting and the Program Generator.

Select "Main Menu 4) Source Code Editor" and load CUSTPYMT. Answer "Y" to the question "New file?" The computer will present you with a blank working area. Between the menu and the working area is the **status line.** Read the line to find that @CUSTPYMT.SRR is opened, we are presently on line one, total lines equal one, and insert is off.

Creating the **CUSTPYMT** program --

CUSTPYMT.SRR source file listing

```
          remark
          remark OPEN FILES:  CUSTFILE AND CUSTPYMT
          trap SAVE goto SAVE.GRP
          trap DELETE goto DELT.GRP
          mount screen CUSTPYMT
START     enter PYMT.CUSCODE
          find rec using field PYMT.CUSCODE related field CUS.CUSCODE
                                                  on error goto START
          enter PYMT.DATE
          enter PYMT.PAYMENT
          CUS.RECEIVE = (CUS.RECEIVE)-(PYMT.PAYMENT)
SAVE.GRP  save rec in file CUSTPYMT confirm / clear buffer
          save rec in file CUSTFILE no confirm / clear buffer
          goto START
DELT.GRP  delete rec in file CUSTPYMT confirm
          goto START
```

Turn the insert on by pressing **[I]**. You are now in the insert mode and will continue so until you return to this menu and change to a different mode.

A ten character long label prompt (10 *****'s) will be displayed in the upper left hand corner of your screen. If you anticipate that the command you are about to write will be branched to from other places in your program, this is the space provided to enter the "target" label to which control may be diverted. We do not want a label on this line so press **[RETURN]**.

SENSIBLE SOLUTION automatically opens files as they are needed. However, for information purposes, at the beginning of your source code file it's a good idea to note the files used by your program

Press **[!]** for remark

You will be presented with a row of 65 ***'s.** This space is provided for you to enter whatever remark you wish. The remark will have no effect while executing programs.

Type **OPEN FILES: CUSTFILE AND CUSTPYMT [RETURN]**

The computer will then ask: Save this new line? Yes No

Press **[Y]** or **[RETURN]**

The line that you just wrote appears in your program as:

```
remark OPEN FILES:  CUSTFILE AND CUSTPYMT
```

The next thing we want to do is to set **"traps"** for certain conditions (Save record and Remove record). If at any point during execution of this program the program operator presses **[ESC] [S]** or **[ESC] [R]** to save or remove a record, we want program control to jump to another portion of the program where the save and delete record commands will be executed.

Our next command line will look like this:

trap SAVE goto SAVE.GRP

Now let's create it.

Again you will be presented with a label input area.

Press **[RETURN]** no label

Press **[T]** for trap

Creating the **CUSTPYMT** program --

CUSTPYMT.SRR source file listing

```
           remark
           remark OPEN FILES:  CUSTFILE AND CUSTPYMT
           trap SAVE goto SAVE.GRP
           trap DELETE goto DELT.GRP
           mount screen CUSTPYMT
START      enter PYMT.CUSCODE
           find rec using field PYMT.CUSCODE related field CUS.CUSCODE
                                                         on error goto START
           enter PYMT.DATE
           enter PYMT.PAYMENT
           CUS.RECEIVE = (CUS.RECEIVE)-(PYMT.PAYMENT)
SAVE.GRP   save rec in file CUSTPYMT confirm / clear buffer
           save rec in file CUSTFILE no confirm / clear buffer
           goto START
DELT.GRP   delete rec in file CUSTPYMT confirm
           goto START
```

Press **[S]**	for Save
Press **[G]**	for Goto or sub
Type **SAVE.GRP [RETURN]**	as the label to jump to
Press **[RETURN]**	to save the line

Now on to the next line:

trap DELETE goto DELT.GRP

Press **[RETURN]**	no label
Press **[T]**	for Trap
Press **[D]**	for Delete
Press **[G]**	for Goto or sub
Type **DELT.GRP [RETURN]**	as the label to jump to
Press **[RETURN]**	to save the line

The next thing the program should do is display the screen that we painted earlier. So we will write the command:

mount screen CUSTPYMT

Press **[RETURN]**	no label
Press **[M]**	for Mount
Press **[S] or [RETURN]**	for screen format
Type **[CUSTPYMT]**	for the screen name
Press **[RETURN]**	to save the line

We want to mark the actual beginning of the program data input and start allowing data entry. Our command line will look like this:

START enter PYMT.CUSCODE

Creating the **CUSTPYMT** program --

CUSTPYMT.SRR source file listing

```
0001            remark
0002            remark OPEN FILES:  CUSTFILE AND CUSTPYMT
0003            trap SAVE goto SAVE.GRP
0004            trap DELETE goto DELT.GRP
0005            mount screen CUSTPYMT
0006 START      enter PYMT.CUSCODE
0007            find rec using field PYMT.CUSCODE related field CUS.CUSCODE
                                                         on error goto START
0008            enter PYMT.DATE
0009            enter PYMT.PAYMENT
0010            CUS.RECEIVE = (CUS.RECEIVE)-(PYMT.PAYMENT)
0011 SAVE.GRP   save rec in file CUSTPYMT confirm / clear buffer
0012            save rec in file CUSTFILE no confirm / clear buffer
0013            goto START
0014 DELT.GRP   delete rec in file CUSTPYMT confirm
0015            goto START
```

This time we want to label the command line using the label field (10 *'s) at the upper left corner of the screen.

Type **START [RETURN]**	command line label
Press **[E]**	for enter
Type **PYMT.CUSCODE [RETURN]**	for field name
Press **[RETURN]**	no mask
Press **[RETURN]**	for no password
Press **[RETURN]**	for no on help gosub
Press **[RETURN]**	for no on up arrow goto
Press **[RETURN]**	to save the line

The two files we are using in our program, **CUSTFILE** and **CUSTPYMT,** contain different information but they are **"related"** by the account-number field. Every master customer record will have a unique account number; but there will probably be several payment records for each customer. We want SENSIBLE SOLUTION to retrieve the appropriate customer record and to display name, receivables, and sales-year-to-date when we enter a new customer payment record.

To do this, we have to find the "related" record. We'll insert a command to locate the record "related" to **PYMT.CUSCODE.**

Our command line will read:

find rec using field PYMT.CUSCODE related field CUS.CUSCODE on error got START

Enter the following:

[RETURN]	no label
[F]	for Find
[R]	for Related record
PYMT.CUSCODE [RETURN]	For related field to use
CUS.CUSCODE [RETURN]	for related field to find

Creating the **CUSTPYMT** program --

CUSTPYMT.SRR source file listing

```
          remark
          remark OPEN FILES:  CUSTFILE AND CUSTPYMT
          trap SAVE goto SAVE.GRP
          trap DELETE goto DELT.GRP
          mount screen CUSTPYMT
START     enter PYMT.CUSCODE
          find rec using field PYMT.CUSCODE related field CUS.CUSCODE
                                                   on error goto START
          enter PYMT.DATE
          enter PYMT.PAYMENT
          CUS.RECEIVE = (CUS.RECEIVE)-(PYMT.PAYMENT)
SAVE.GRP  save rec in file CUSTPYMT confirm / clear buffer
          save rec in file CUSTFILE no confirm / clear buffer
          goto START
DELT.GRP  delete rec in file CUSTPYMT confirm
          goto START
```

START [RETURN]	if an error occurs (e.g., can't find the related field) go to this label
[RETURN]	to save this new line

Now let's write two more "enter" commands, allowing data input to our program. The first will say:

enter PYMT.DATE

Enter the following:

[RETURN]	no label
[E]	for enter
PYMT.DATE [RETURN]	field name
[RETURN]	for no mask
[RETURN]	for no password
[RETURN]	for no on help gosub
[RETURN]	for no on up arrow goto
[RETURN]	to save the new line

The next "enter" command is:

enter PYMT.PAYMENT

Enter the following:

[RETURN]	no label
[E]	for enter
PYMT.PAYMENT [RETURN]	field name
[RETURN]	no mask
[RETURN]	for no password
[RETURN]	for no on help gosub

Creating the **CUSTPYMT** program --

CUSTPYMT.SRR source file listing

```
           remark
           remark OPEN FILES:  CUSTFILE AND CUSTPYMT
           trap SAVE goto SAVE.GRP
           trap DELETE goto DELT.GRP
           mount screen CUSTPYMT
START      enter PYMT.CUSCODE
           find rec using field PYMT.CUSCODE related field CUS.CUSCODE
                                                    on error goto START
           enter PYMT.DATE
           enter PYMT.PAYMENT
           CUS.RECEIVE = (CUS.RECEIVE)-(PYMT.PAYMENT)
SAVE.GRP   save rec in file CUSTPYMT confirm / clear buffer
           save rec in file CUSTFILE no confirm / clear buffer
           goto START
DELT.GRP   delete rec in file CUSTPYMT confirm
           goto START
```

[RETURN]	for no on up arrow goto
[RETURN]	to save the new line

We want to reduce **receivables** in the CUSTFILE master-customer record by the amount of the **payment.** Our program must compute the new amount of outstanding receivables. We'll insert a command line after the line allowing payment entry to calculate "receivables minus payment" and store the result back into "receivables."

The command line will read:

CUS.RECEIVE = (CUS.RECEIVE) - (PYMT.PAYMENT)

Enter the following:

[RETURN]	no label
[=]	to designate some sort of computation
CUS.RECEIVE [RETURN]	the destination field where the answer will be placed
[E]	for Expression

In the SENSIBLE SOLUTION, a **calc expression** calculates a new value using the current values in fields, or constant numeric or constant alphanumeric data. The command [=] takes this newly-calculated value and places it in the field.

Complex algebraic expressions seldom occur in business applications so the SENSIBLE SOLUTION uses a very simple systems for expression calculations.

Expressions may contain the current value of a field, constants like the number 3.50, and alphanumeric strings like ABCGOAT>

The standard arithmetic operations are indicated by:

+ (addition) - (subtraction) * (multiplication) / (division)

The kind of data is indicated by the type of brackets around it:

(CUS.CUSCODE)	( ) marks a field name
<4500.50>	< > marks a numeric constant
[Account No.]	[ ] marks an alphanumeric string constant

Creating the **CUSTPYMT** program --

CUSTPYMT.SRR source file listing

```
          remark
          remark OPEN FILES:  CUSTFILE AND CUSTPYMT
          trap SAVE goto SAVE.GRP
          trap DELETE goto DELT.GRP
          mount screen CUSTPYMT
START     enter PYMT.CUSCODE
          find rec using field PYMT.CUSCODE related field CUS.CUSCODE
                                                     on error goto START
          enter PYMT.DATE
          enter PYMT.PAYMENT
          CUS.RECEIVE = (CUS.RECEIVE)-(PYMT.PAYMENT),
SAVE.GRP  save rec in file CUSTPYMT confirm / clear buffer
          save rec in file CUSTFILE no confirm / clear buffer
          goto START
DELT.GRP  delete rec in file CUSTPYMT confirm
          goto START
```

The appropriate brackets **must** be used so that the characters **"+"**, **"-"**, **"*"** and **"/"** may be used as part of a string or a field name. Remember, with SENSIBLE SOLUTION, it's entirely legal! **"+"** may **also** indicate **string concatenation**, which we'll talk about later.

Continue entering:

(CUS.RECEIVE)-(PYMT.PAYMENT) [RETURN]	the calculated expression
[RETURN]	to save the new line

The command that we have just created updates the "receivables" field in our Customer Master File by subtracting the amount of the payment. Of course, the customer file is not actually updated until the customer record is saved back to the disk. We need two commands--a "save" to the payments file and a "save" to the Customer Master File.

Our next command is:

save rec in file CUSTPYMT confirm/clear buffer

Enter the following:

SAVE.GRP [RETURN]	command line label
[S]	for Save rec
CUSTPYMT	file name
[Y]	so the save must be confirmed before happens
[RETURN]	yes to clear the buffer of the record
[RETURN]	No "goto" line label is needed. We want program flow to continue with the next sequential command. If you wished program flow to branch to an alternative command, you would enter that command line label.
[RETURN]	save this new line

Our second "save" command looks like this:

save rec in file CUSTFILE no confirm/clear buffer

Creating the **CUSTPYMT** program --

CUSTPYMT.SRR source file listing

```
           remark
           remark OPEN FILES:  CUSTFILE AND CUSTPYMT
           trap SAVE goto SAVE.GRP
           trap DELETE goto DELT.GRP
           mount screen CUSTPYMT
START      enter PYMT.CUSCODE
           find rec using field PYMT.CUSCODE related field CUS.CUSCODE
                                                   on error goto START
           enter PYMT.DATE
           enter PYMT.PAYMENT
           CUS.RECEIVE = (CUS.RECEIVE)-(PYMT.PAYMENT)
SAVE.GRP   save rec in file CUSTPYMT confirm / clear buffer
           save rec in file CUSTFILE no confirm / clear buffer
           goto START
DELT.GRP   delete rec in file CUSTPYMT confirm
           goto START
```

Enter the following:

[RETURN]	no label
[S]	for Save rec
CUSTFILE	file name
[N]	so the save need not be confirmed before it happens
[RETURN]	yes to clear the buffer of the record
[RETURN]	save this new line

Now create the command line:

goto START

Enter the following:

[RETURN]	no label
[G]	for Go command
[T]	for goTo option
[RETURN]	not dependent on field value
START [RETURN]	line label to branch to
[RETURN]	save new line

The next line is:

delete rec in file CUSTPYMT confirm

Enter the following:

DELT.GRP [RETURN]	line label
[D]	for delete record
CUSTPYMT	file name
[RETURN]	a confirm must be answered before delete

Creating the **CUSTPYMT** program --

CUSTPYMT.SRR source file listing

```
           remark
           remark OPEN FILES:  CUSTFILE AND CUSTPYMT
           trap SAVE goto SAVE.GRP
           trap DELETE goto DELT.GRP
           mount screen CUSTPYMT
START      enter PYMT.CUSCODE
           find rec using field PYMT.CUSCODE related field CUS.CUSCODE
                                                   on error goto START
           enter PYMT.DATE
           enter PYMT.PAYMENT
           CUS.RECEIVE = (CUS.RECEIVE)-(PYMT.PAYMENT)
SAVE.GRP   save rec in file CUSTPYMT confirm / clear buffer
           save rec in file CUSTFILE no confirm / clear buffer
           goto START
DELT.GRP   delete rec in file CUSTPYMT confirm
           goto START
```

[RETURN]	no label needed
[RETURN]	save this new line

Our final command line is:

goto START

Enter the following:

[RETURN]	no label
[G]	for Go
[T]	for goTo
[RETURN]	no dependent on field value
START [RETURN]	line label to branch to
[RETURN]	save new line

Your completed program should look like the one on the opposite page.

Did you make any mistakes? If you find that you want to delete a line of your source code, press **[ESC]** to call up the Source Code Editor menu. Use the **[UP ARROW]** and **[DOWN ARROW]** keys to position the line you want deleted next to the **">"** prompt (or use **[ESC] [F]** Find line). Select "Delete line" option and answer the question

Do you wish to delete this line? Yes No

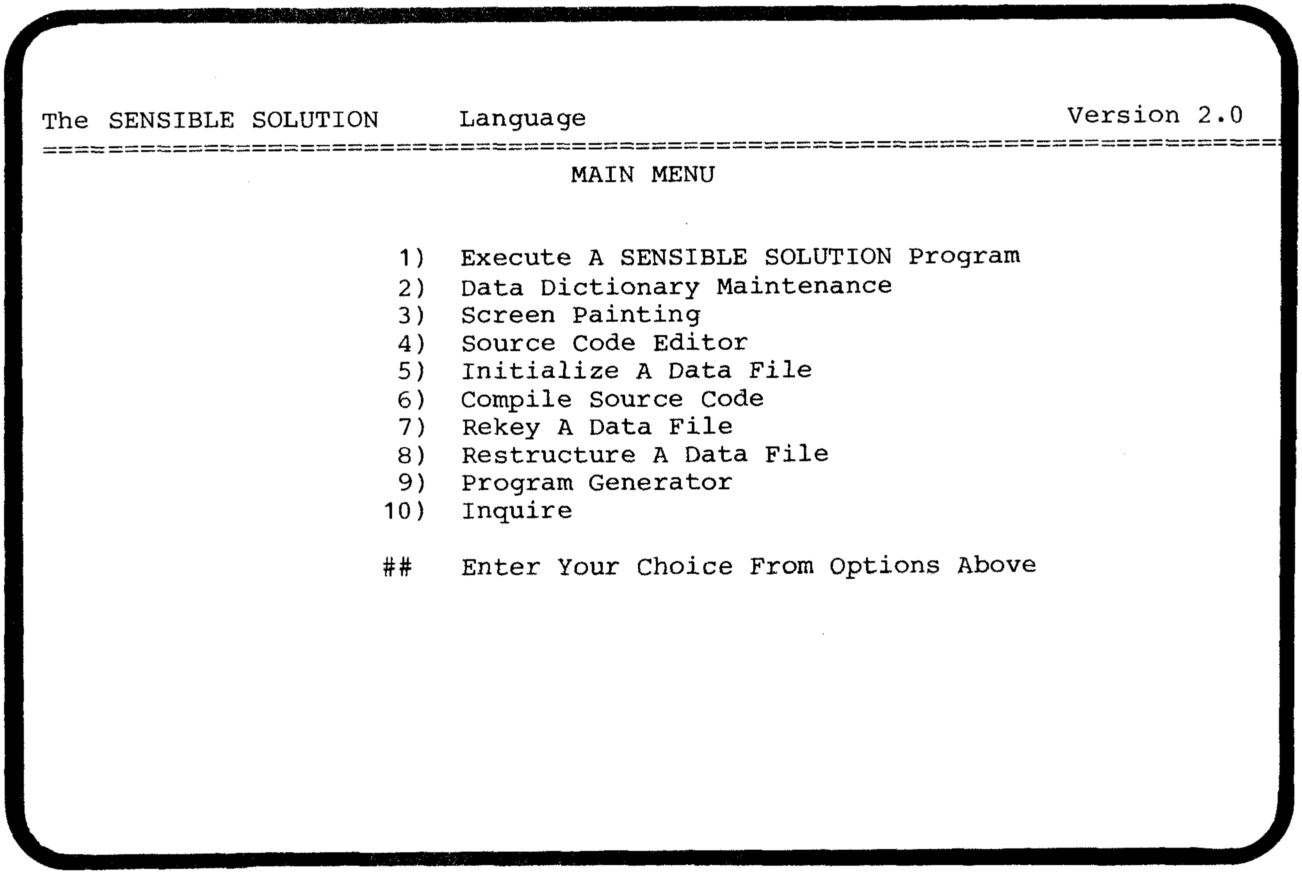
Now simply "Insert" the proper line, using the "Insert line" option.

If you want to change a part of a line in your code, position the line next to the **">"** prompt and select the "Change line" option. Press **[RETURN]** and the current command options will be displayed one step at a time. You can change any of the entries, or change the entire command.

Take some time and experiment with these options. After a little practice you'll begin to appreciate how easy it is to use the SENSIBLE SOLUTION!

To exit out of the insert mode press your **[ESC]** key. This will bring you back to the Source Code Editor menu. Press **[Q]** for Quit. You will be asked:

After pressing **[ESC]** **[Q]** (Quit) --



```
The SENSIBLE SOLUTION     Language                                Version 2.0
=============================================================================
                                 MAIN MENU


                      1)  Execute A SENSIBLE SOLUTION Program
                      2)  Data Dictionary Maintenance
                      3)  Screen Painting
                      4)  Source Code Editor
                      5)  Initialize A Data File
                      6)  Compile Source Code
                      7)  Rekey A Data File
                      8)  Restructure A Data File
                      9)  Program Generator
                     10)  Inquire

                     ##   Enter Your Choice From Options Above
                     +---+
Exit completely?     |Yes|   No
                     +---+
```

Press **[RETURN]**.

Now you must compile the source code file. Compilation is the process of creating pseudo code that the SENSIBLE SOLUTION executive program can run. Select "Main Menu 6) Compile Source Code" and specify **CUSTPYMT** as the file to be compiled.

The compiler will scan the source code listing and verify all labels, screens, fields, and files. Finally, the compiler will create the command file CUSTPYMT.RUN which is used by the SENSIBLE SOLUTION executive (Main Menu option 1). When the compiler is finished working, it returns you to the Main Menu.

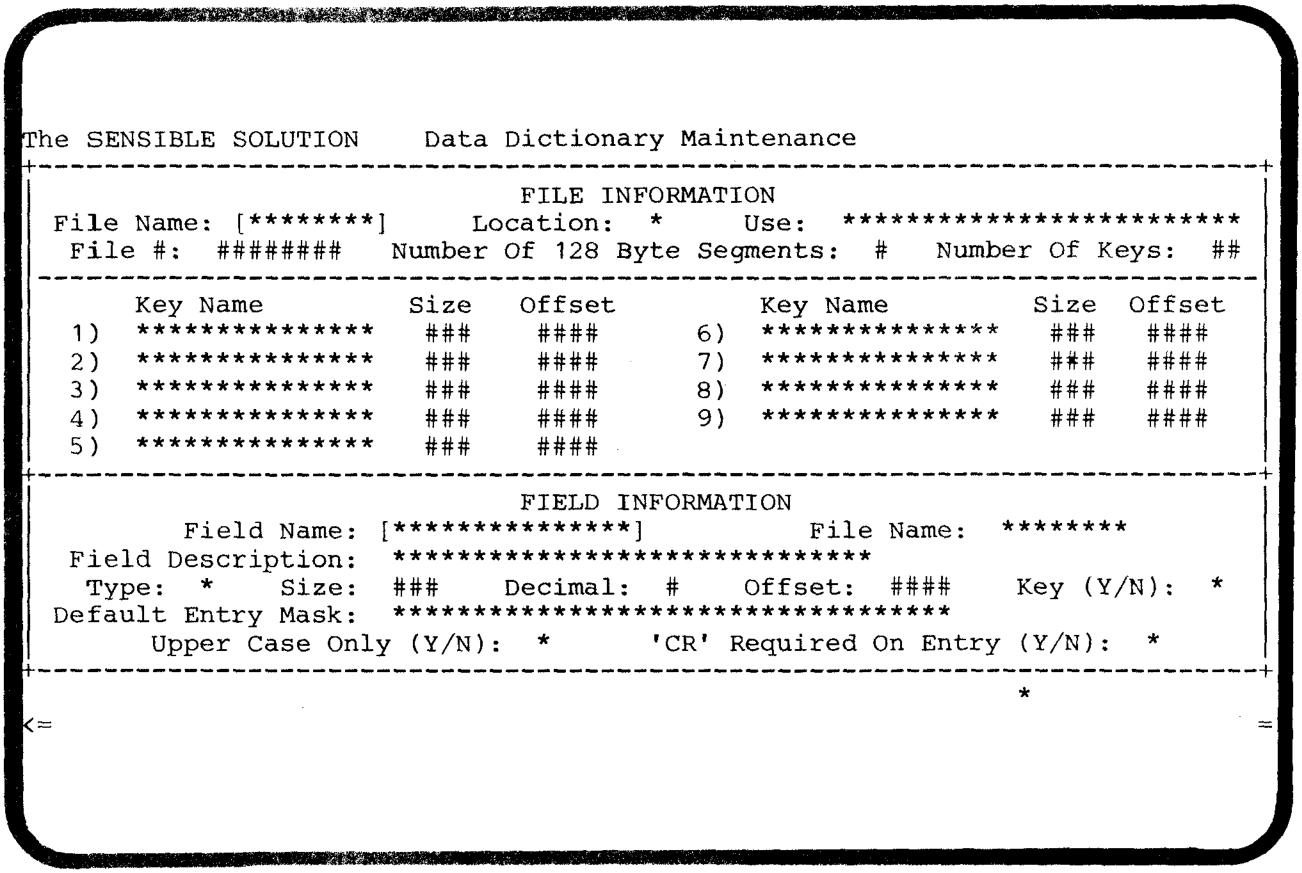
Now let's try out our new program. Select "Main Menu 1) Execute a SENSIBLE SOLUTION Program" and enter the file name **CUSTPYMT.**

Enter account number 2222 and press **[RETURN]**. "2222" was one of the customer entries you entered with your first screen. The customer's name will automatically appear.

Now enter a date for the "payment" and then **457.65** for the amount of the payment. Take a look at the outstanding receivables" window on your screen. You'll see that **-457.65** has appeared in it. This is the credit balance the customer has for his payment of $457.65 against a zero receivables total. Press **[Y]** or **[RETURN]** to store this payment record in the **CUSTPYMT** data file and to save the updated value for Receivables in the Customer Master File data file record (CUSTFILE).

You'll have a hard time talking your customer into making payments only! Let's create a sales-order screen.

The Data Dictionary maintenance screen --



```
The SENSIBLE SOLUTION    Data Dictionary Maintenance
+----------------------------------------------------------------------------+
|                              FILE INFORMATION                              |
| File Name: [********]     Location:  *     Use:  *************************  |
|  File #:  ########   Number Of 128 Byte Segments:  #   Number Of Keys:  ##  |
|----------------------------------------------------------------------------|
|      Key Name        Size   Offset         Key Name        Size  Offset    |
|  1)  ***************   ###    ####     6)  ***************   ###   ####    |
|  2)  ***************   ###    ####     7)  ***************   ###   ####    |
|  3)  ***************   ###    ####     8)  ***************   ###   ####    |
|  4)  ***************   ###    ####     9)  ***************   ###   ####    |
|  5)  ***************   ###    ####                                          |
+----------------------------------------------------------------------------+
|                              FIELD INFORMATION                             |
|        Field Name: [***************]         File Name:  ********          |
|  Field Description:  ******************************                        |
|   Type:  *    Size:  ###    Decimal:  #    Offset:  ####    Key (Y/N):  *   |
| Default Entry Mask:  ***********************************                   |
|       Upper Case Only (Y/N):  *      'CR' Required On Entry (Y/N):  *      |
+----------------------------------------------------------------------------+
                                                              *
<=                                                                          =
```

LESSON 6
Defining Data Files With The Data Dictionary

Now that you understand how the Screen Painter, Source Code Editor, and Program Generator work, we are going to take another approach to creating a program. This time we'll begin the whole process by setting up the file and fields in the Data Dictionary. Then we'll paint the screen with "Main Menu 3) Screen Painting" and create the program with the Source Code Editor.

Data Dictionary Maintenance is a SENSIBLE SOLUTION program used to update four SENSIBLE SOLUTION data files, **FLDFLE.MS/.KS** and **RECFLE.MS/.KS.** These four files comprise the Data Dictionary. When a SENSIBLE SOLUTION program is run, it accesses the Data Dictionary (RECFLE.MS) to determine the drive locations of your data files (filename.MS/.KS). In addition to this file information, the Data Dictionary contains a complete definition of every field used in a SENSIBLE SOLUTION program. Before, we used Screen Painting to create or "define" fields. But don't be confused, the field definitions that you created while you were painting a screen went into the Data Dictionary. This time, however, we're going to go directly into the Data Dictionary to create some new field definitions.

Select "Main Menu 2) Data Dictionary Maintenance." You'll see a display like the one shown on the opposite page. We'll begin by examining the field definitions of the master data file we have already created. Enter the file name, **CUSTFILE**, in the first field window on the Data Dictionary screen and press **[RETURN]**.

Each field window on this screen displays information about the file and the fields that comprise the file:

File Name: The disk-directory file name, user-specified.

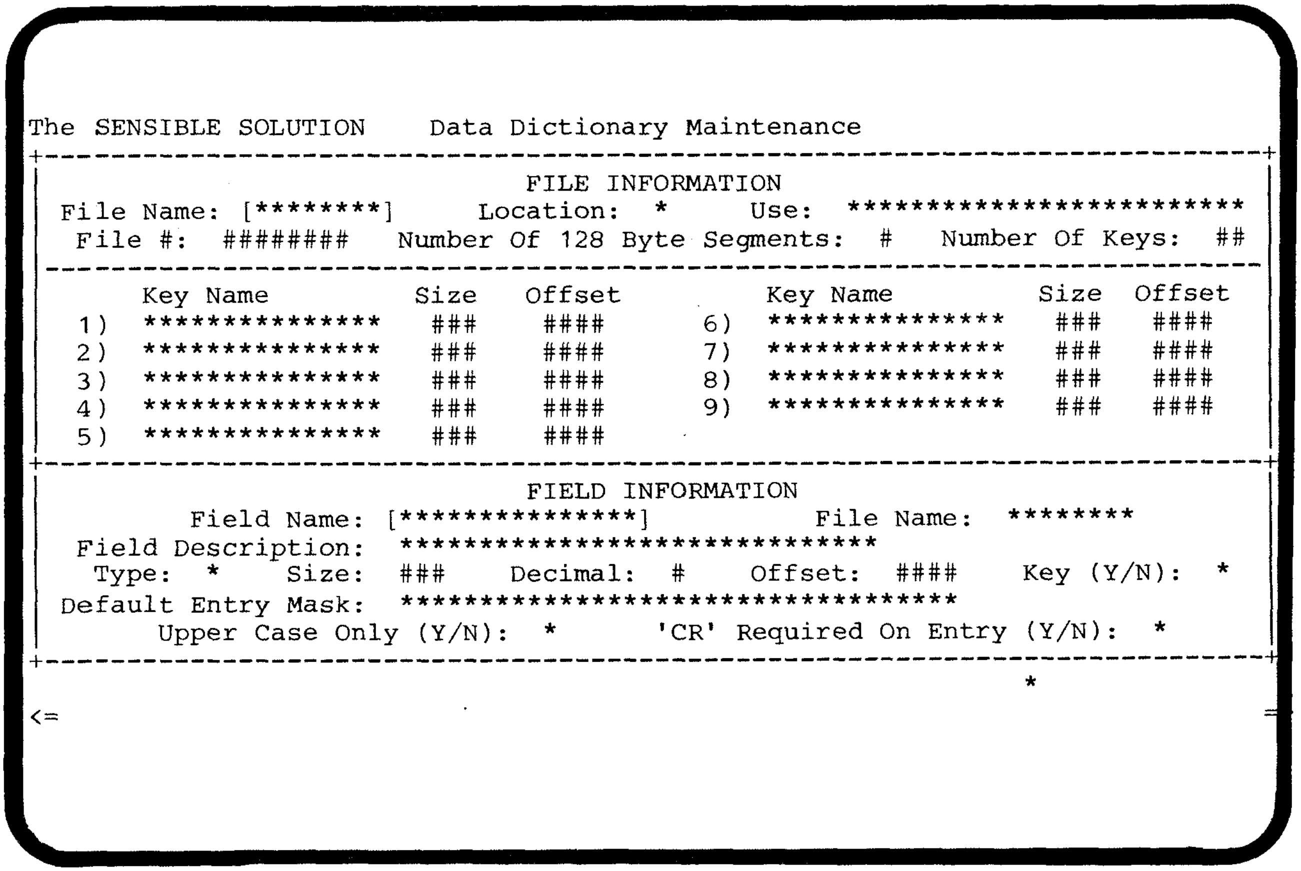
Location: Which disk drive the file resides on.

Use: Designer enters a "comment" about the file here.

File Num: The SENSIBLE SOLUTION uses this number to track files in use.

Number of 128 Byte Segments: Currently defined size of data record expressed in 128 byte units.

The Data Dictionary maintenance screen --



```
The SENSIBLE SOLUTION    Data Dictionary Maintenance
+----------------------------------------------------------------------------+
|                           FILE INFORMATION                                 |
| File Name: [********]    Location:  *     Use:   ************************* |
|  File #:  ########  Number Of 128 Byte Segments:  #   Number Of Keys:  ##  |
|----------------------------------------------------------------------------|
|      Key Name            Size   Offset          Key Name          Size  Offset
|  1)  ***************     ###    ####      6)  ***************    ###   ####
|  2)  ***************     ###    ####      7)  ***************    ###   ####
|  3)  ***************     ###    ####      8)  ***************    ###   ####
|  4)  ***************     ###    ####      9)  ***************    ###   ####
|  5)  ***************     ###    ####                                     |
+----------------------------------------------------------------------------+
|                           FIELD INFORMATION                                |
|        Field Name: [***************]          File Name:  ********        |
|  Field Description:  ******************************                        |
|   Type:  *    Size:  ###    Decimal:  #    Offset:  ####    Key (Y/N):  * |
| Default Entry Mask:  ***********************************                   |
|       Upper Case Only (Y/N):  *      'CR' Required On Entry (Y/N):  *     |
+----------------------------------------------------------------------------+
                                                                   *
<=                                                                           =
```

Number of Keys:	Number of index keys used in this file.
Key Name:	Field name of field used for a key.
Size:	Number of bytes in key field.
Offset:	Beginning byte position of key in record.

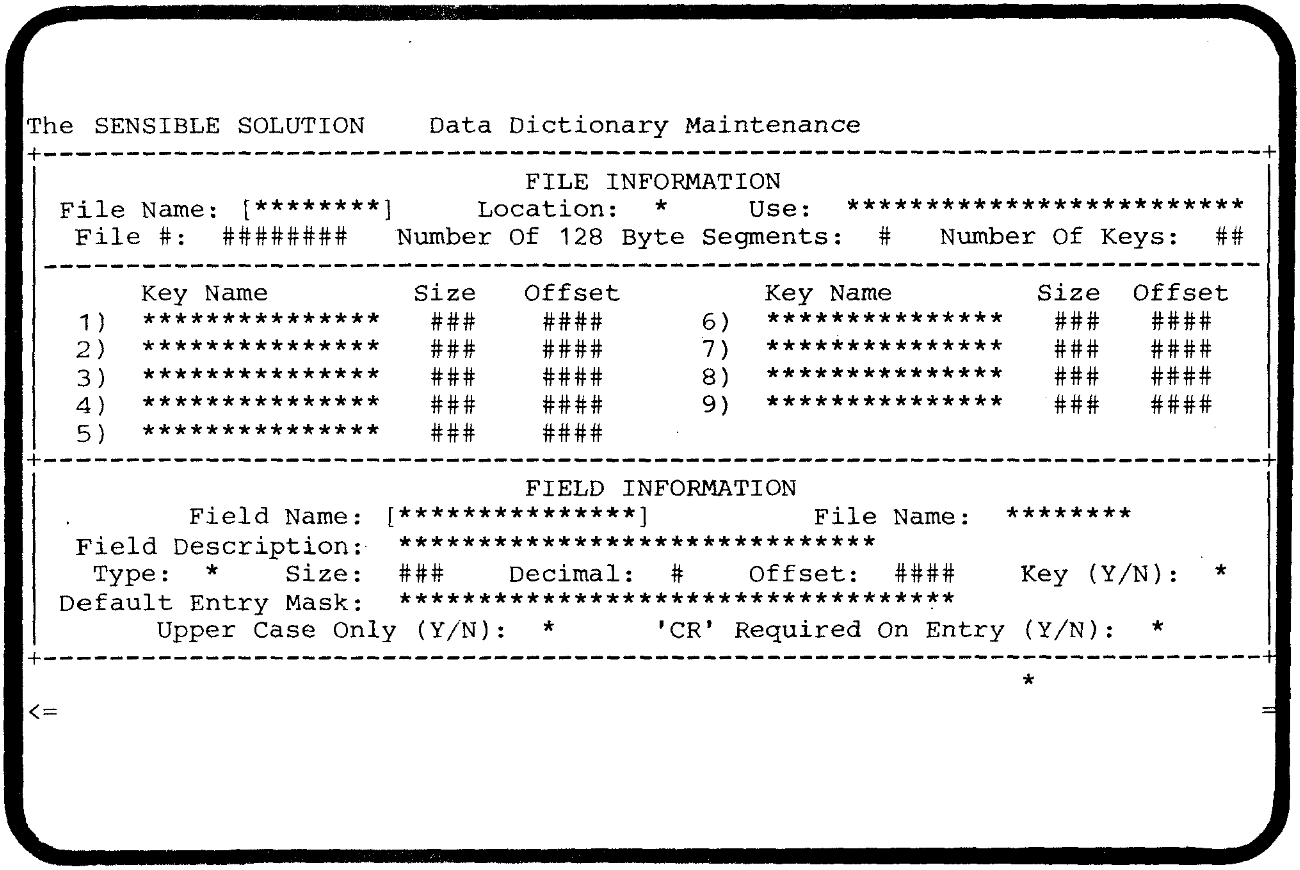
You can change any of the displayed information on the screen, call up other file and field definitions, or create brand new file and field definitions in the Data Dictionary all with this one screen. One point we should note here, Data Dictionary Maintenance is itself a SENSIBLE SOLUTION program. Because of this, you can page through the files using any of the standard program control keys to search for records, clear the screen, remove records, etc.. Remember, if you change any of the displayed information on the Data Dictionary screen, the definition in the Data Dictionary will not actually be changed until you save the screen information with **[ESC] [S]**. For detailed information on the operation of **2) Data Dictionary Maintenance**, you should read the section on the Data Dictionary in the Reference Manual.

Now let's look at the Data Dictionary from the perspective of compiling source code, initializing a data file, and executing a SENSIBLE SOLUTION program. When you use "Main Menu 6) Compile Source Code the compiler actually goes to the Data Dictionary where it "reads" the definitions of every field that is specified in the source code. This field definition information is then embedded into the pseudo code that the compiler generates (filename.RUN). When the program is finally executed using "Main Menu 1) Execute A SENSIBLE SOLUTION Program, SENSIBLE SOLUTION "knows" the exact definition of every field it is about to utilize.

When you use "Main Menu 5) Initialize A Data File", the Initialize program also "reads" the Data Dictionary. Initialize is the program that opens up the new data files (filename.MS and filename.KS) on your disk drive so that they will be prepared to receive data from your SENSIBLE SOLUTION program. The field definitions from the Data Dictionary all have to be laid out in advance before a program can be executed.

When you use "Main Menu 1) Execute A SENSIBLE SOLUTION Program", this "executive" program must go to the Data Dictionary to determine the disk drive location of the appropriate data files. It does this every time that you run a SENSIBLE SOLUTION program. Therefore, any time you wish, you can use a copy utility such as PIP to move your data files (filename.MS/.KS) to a different drive location then go back to the Data Dictionary and change the "Location" field window to reflect the new drive location. In this way you can locate the data files anywhere you want on your system and

The Data Dictionary maintenance screen --



```
The SENSIBLE SOLUTION    Data Dictionary Maintenance
+----------------------------------------------------------------------------+
|                              FILE INFORMATION                              |
| File Name: [********]     Location:  *     Use:  ************************* |
|  File #:  ########   Number Of 128 Byte Segments:  #   Number Of Keys:  ## |
|----------------------------------------------------------------------------|
|      Key Name            Size   Offset         Key Name            Size  Offset|
|  1)  ***************     ###    ####     6)  ***************     ###   #### |
|  2)  ***************     ###    ####     7)  ***************     ###   #### |
|  3)  ***************     ###    ####     8)  ***************     ###   #### |
|  4)  ***************     ###    ####     9)  ***************     ###   #### |
|  5)  ***************     ###    ####                                       |
+----------------------------------------------------------------------------+
|                              FIELD INFORMATION                             |
|        Field Name: [***************]          File Name:  ********         |
|  Field Description: ******************************                         |
|   Type:  *    Size:  ###    Decimal:  #    Offset:  ####    Key (Y/N):  *  |
| Default Entry Mask: ***********************************                    |
|       Upper Case Only (Y/N):  *      'CR' Required On Entry (Y/N):  *     |
+----------------------------------------------------------------------------+
                                                             *
<=                                                                          =
```

everything will work just fine.

So you can see, the Data Dictionary is really the core of the SENSIBLE SOLUTION. All information used by a SENSIBLE SOLUTION program must, one way or another, pass through the Data Dictionary.

Now let's return to creating the sales order program, CUSTORD. Clear the fields in the **FILE INFORMATION** portion of the screen by pressing **[ESC] [C]**. Your cursor should be positioned at the first character of the "File Name" field window. Type **CUSTORD** and **[RETURN]** and then press the **[RETURN]** key again to accept the default drive in the "Location" field window. In the "Use" field window, type in **RECORD CUSTOMER ORDERS**. Press **[ESC] [S]** to save this screen of information and then press **[ESC] [J]** to activate the "Jump screen" option. The cursor will now move to the **FIELD INFORMATION** portion of this screen.

The "jump screen" option is a common practice employed in SENSIBLE SOLUTION programs to give the computer operator the option of using **[ESC] [J]** to jump from one screen to another in one direction and then use **[ESC] [Q]** to jump back the other direction. Don't forget, the Data Dictionary is a SENSIBLE SOLUTION program.

Your cursor should be positioned at the first character of the "Field Name" field window. Press **[ESC] [C]** to clear the fields. We will now define all the fields that will be used by the CUSTORD program that have not already been created.

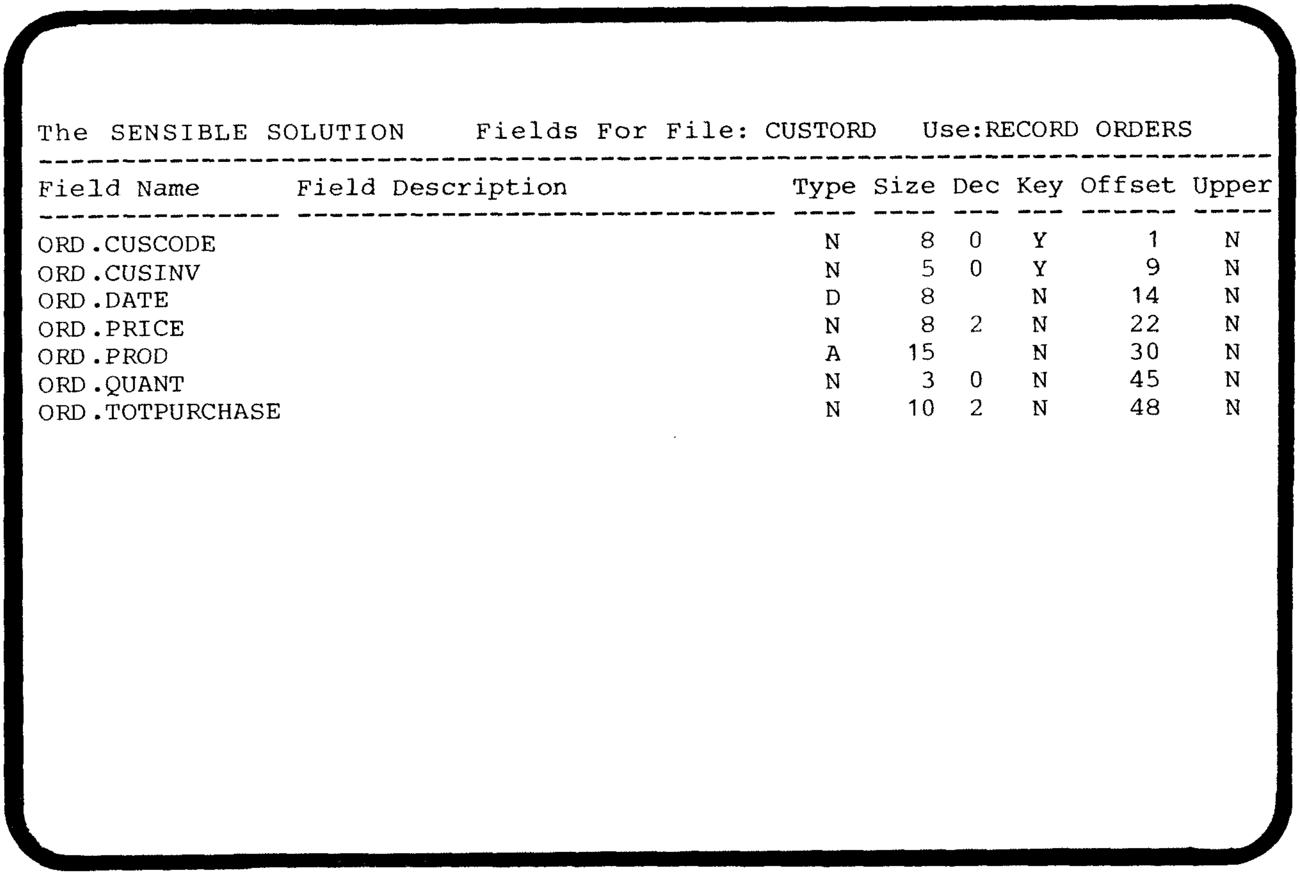
Enter the following information in the appropriate fields:

Field name:	ORD.CUSCODE
File name:	CUSTORD
Field type:	N
Size:	8
Decimal:	0
Key:	Y

Press **[ESC]** and then save the record. Proceed with the following entries:

Field name:	ORD.DATE
File name:	[RETURN]
Field type:	D
Key:	N

Data Dictionary report of CUSTORD sent to CRT --

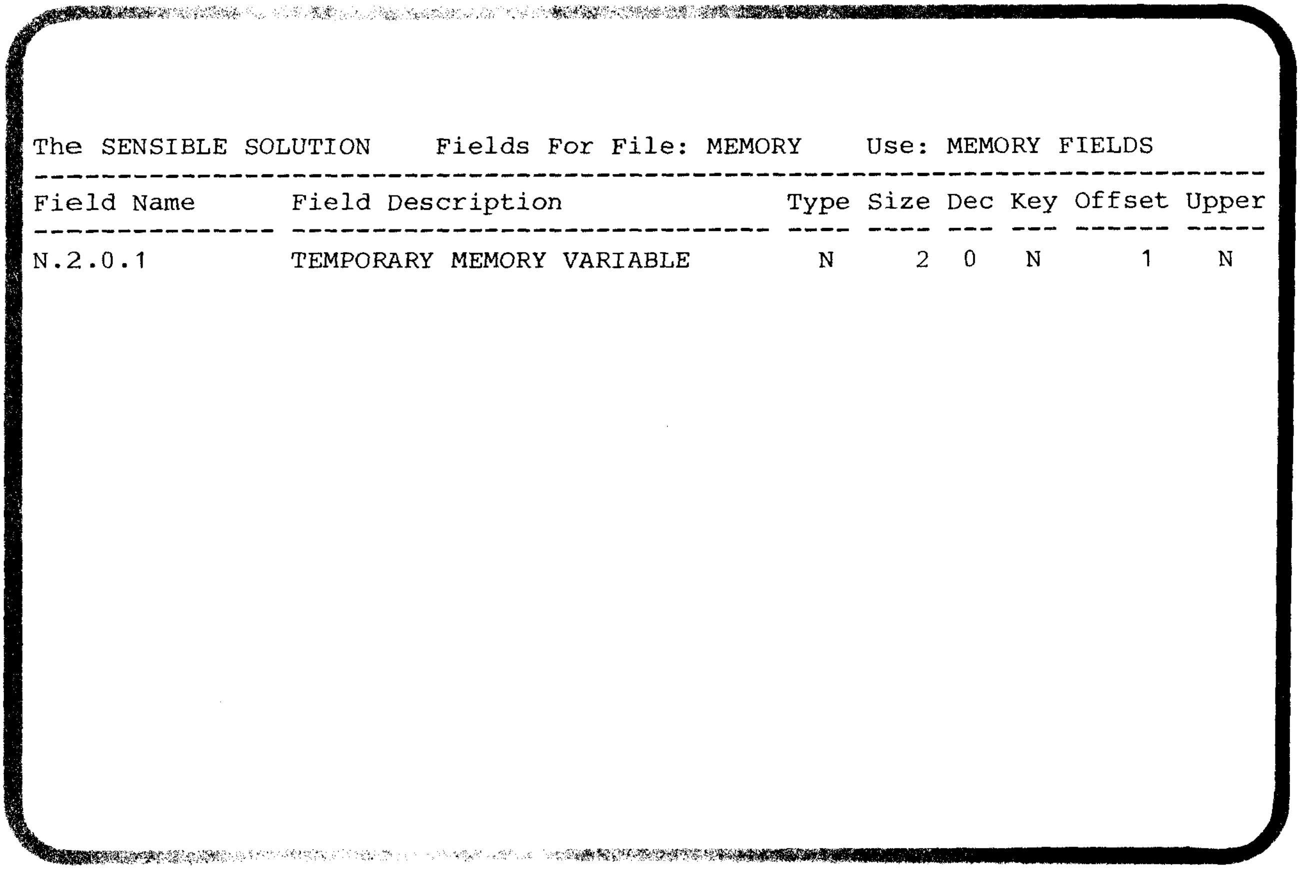


```
The SENSIBLE SOLUTION    Fields For File: CUSTORD   Use:RECORD ORDERS
-----------------------------------------------------------------------------
Field Name       Field Description              Type Size Dec Key Offset Upper
---------------- ------------------------------ ---- ---- --- --- ------ -----
ORD.CUSCODE                                      N      8   0  Y       1   N
ORD.CUSINV                                       N      5   0  Y       9   N
ORD.DATE                                         D      8      N      14   N
ORD.PRICE                                        N      8   2  N      22   N
ORD.PROD                                         A     15      N      30   N
ORD.QUANT                                        N      3   0  N      45   N
ORD.TOTPURCHASE                                  N     10   2  N      48   N
```

```
Field name:    ORD.CUSINV
File name:     [RETURN]
Field type:    N
Size:          5
Decimal:       0
Key:           Y
Field name:    ORD.PROD
File  name:    [RETURN]
Field type:    A
Size:          15
Key:           N

Field name:    ORD.PRICE
File name:     [RETURN]
Field type:    N
Size:          8
Decimal:       2
Key:           N

Field name:    ORD.QUANT
File name:     [RETURN]
Field type:    N
Size:          3
Decimal:       0
Key:           N

Field name:    ORD.TOTPURCHASE
File name:     [RETURN]
Field type:    N
Size:          10
Decimal:       2
Key:           N
```

We also want to create a new kind of field to be used in our CUSTORD program called a "memory" variable. This field, or variable, will be used to hold temporary results of calculations. You'll remember that every field must be assigned to a data file. We'll create a file called **MEMORY**, which will contain only this temporary "holding" field.

Of course, there's no sense wasting disk space on temporary data. So we'll use a trick -- never write a line in a program that saves a **MEMORY** record. **MEMORY.MS** is a real data file, but it contains zero records; hence, it takes up no space on the disk except for an entry in the disk directory. It's a good idea to collect all the temporary fields into a file such as this. We use the MEMORY file to store information like loop counters, Data Dictionary report of MEMORY sent to CRT --



```
The SENSIBLE SOLUTION    Fields For File: MEMORY    Use: MEMORY FIELDS
------------------------------------------------------------------------------
Field Name      Field Description              Type Size Dec Key Offset Upper
--------------- ------------------------------ ---- ---- --- --- ------ -----
N.2.0.1         TEMPORARY MEMORY VARIABLE        N     2  0   N       1   N
```

accumulators, pointers, etc.. These kinds of data values are temporary and are only used internally within the program -- there is never any need to store this kind of data permanently on the disk.

In our CUSTORD program we will need a memory storage field to hold temporary results of calculations. To create MEMORY, do the following:

Press **[ESC] [Q] and [ESC] [C]** to position your cursor at the first character of the "File Name" field window in the FILE INFORMATION portion of the Data Dictionary screen. Now type MEMORY **[RETURN]**. Accept the default on "Location" by pressing **[RETURN]** again, and then define the "Use" as **"TEMP. VARIABLE STORAGE" [RETURN]**. When you are finished save the record with **[ESC] [S]**.

The fields we are about to define in the Data Dictionary will be used to store a number of a month. The naming convention that we'll use for our fields in MEMORY file will look like this:

N.2.0.1

which indicates Numeric, Number of Places, Decimal Places, Entry Number.

S.20.1

means String, Number of Characters, Entry Number

Now let's define a temporary variables in our file MEMORY with this naming convention:

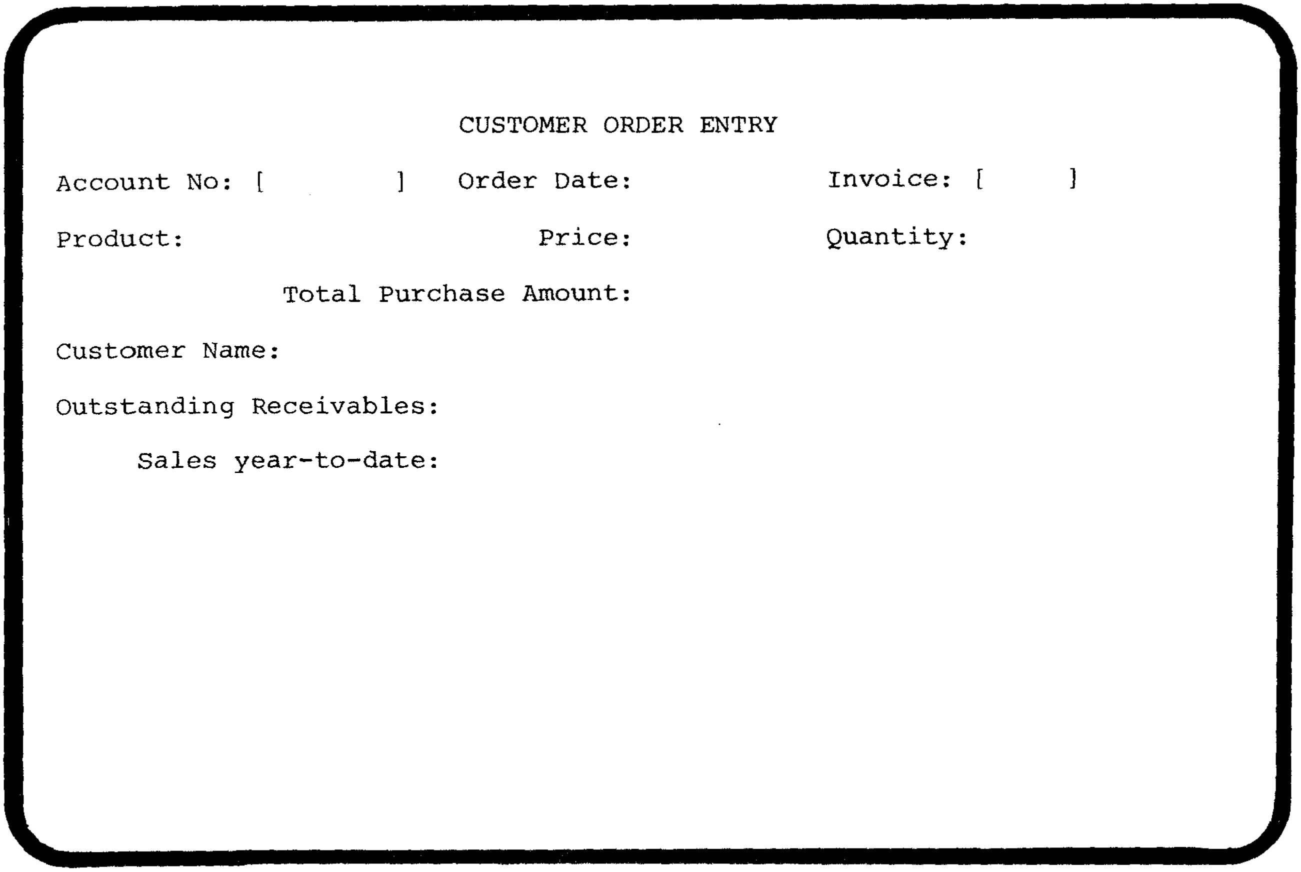
Field Name:	N.2.0.1
File Name:	MEMORY
Type:	N
Size:	2
Decimal:	0
Key?	N

Now press **[ESC] [S]**, and when the computer asks if you want to **SAVE THIS RECORD? (Y/N) Y,** press **Y** to save your new field (or temporary variable) definition.

Before we leave the Data Dictionary screen you should see another one of its features -- the file definition report.

Press **[ESC] [J]** and you will "jump" to a completely different screen where

Paint this CUSTORD screen --



```
                         CUSTOMER ORDER ENTRY

Account No: [        ]   Order Date:           Invoice: [     ]

Product:                      Price:           Quantity:

              Total Purchase Amount:

Customer Name:

Outstanding Receivables:

     Sales year-to-date:
```

accumulators, pointers, etc.. These kinds of data values are temporary and are only used internally within the program -- there is never any need to store this kind of data permanently on the disk.

In our CUSTORD program we will need a memory storage field to hold temporary results of calculations. To create MEMORY, do the following:

Press **[ESC] [Q] and [ESC] [C]** to position your cursor at the first character of the "File Name" field window in the FILE INFORMATION portion of the Data Dictionary screen. Now type MEMORY **[RETURN]**. Accept the default on "Location" by pressing **[RETURN]** again, and then define the "Use" as **"TEMP. VARIABLE STORAGE" [RETURN]**. When you are finished save the record with **[ESC] [S]**.

The fields we are about to define in the Data Dictionary will be used to store a number of a month. The naming convention that we'll use for our fields in MEMORY file will look like this:

N.2.0.1

which indicates Numeric, Number of Places, Decimal Places, Entry Number.

S.20.1

means String, Number of Characters, Entry Number

Now let's define a temporary variables in our file MEMORY with this naming convention:

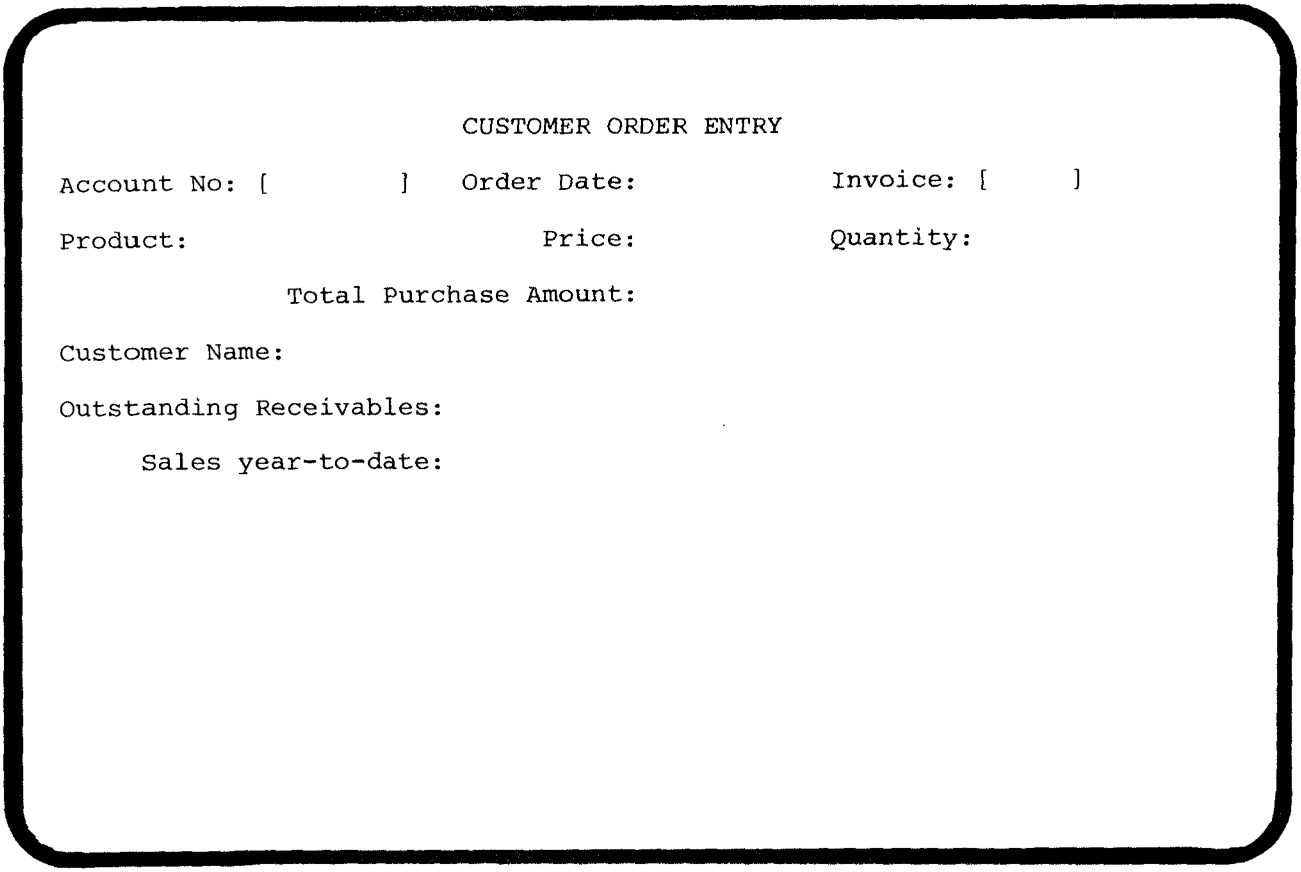
Field Name:	N.2.0.1
File Name:	MEMORY
Type:	N
Size:	2
Decimal:	0
Key?	N

Now press **[ESC] [S]**, and when the computer asks if you want to **SAVE THIS RECORD? (Y/N) Y**, press **Y** to save your new field (or temporary variable) definition.

Before we leave the Data Dictionary screen you should see another one of its features -- the file definition report.

Press **[ESC] [J]** and you will "jump" to a completely different screen where

Paint this CUSTORD screen --



```
                        CUSTOMER ORDER ENTRY

Account No: [        ]  Order Date:            Invoice: [     ]

Product:                     Price:            Quantity:

              Total Purchase Amount:

Customer Name:

Outstanding Receivables:

     Sales year-to-date:
```

you will see the file name **MEMORY** displayed. This screen of the Data Dictionary will allow you to print out a report of all of the field definitions for a specified data file. Just press the **[RETURN]** key and a message will be displayed at the bottom of your screen requesting a destination for the report about to be printed out. For now just answer with a "C" and your screen will display a complete print out of the field definitions for the **MEMORY** data file. In the future you should make frequent use of this feature to review all of your field definitions for all of your files.

Notice that the "File Name" field window on this second screen is a key field. If you press the **[ESC]** key you will see the familiar option menu displayed at the bottom of your screen. So now you can search through the Data Dictionary and bring up any of the previously defined data files and then print out a report of all of your field definitions. Try printing out some reports on your printer of CUSTFILE and CUSTORD.

Now that we're finished with defining the new field in MEMORY, use the **[ESC] [Q]** keys to exit from the Data Dictionary. Before going on with the development of our new program, remember you must **INITIALIZE** the data files for MEMORY and CUSTORD. Use "Main Menu 5) Initialize a Data File.

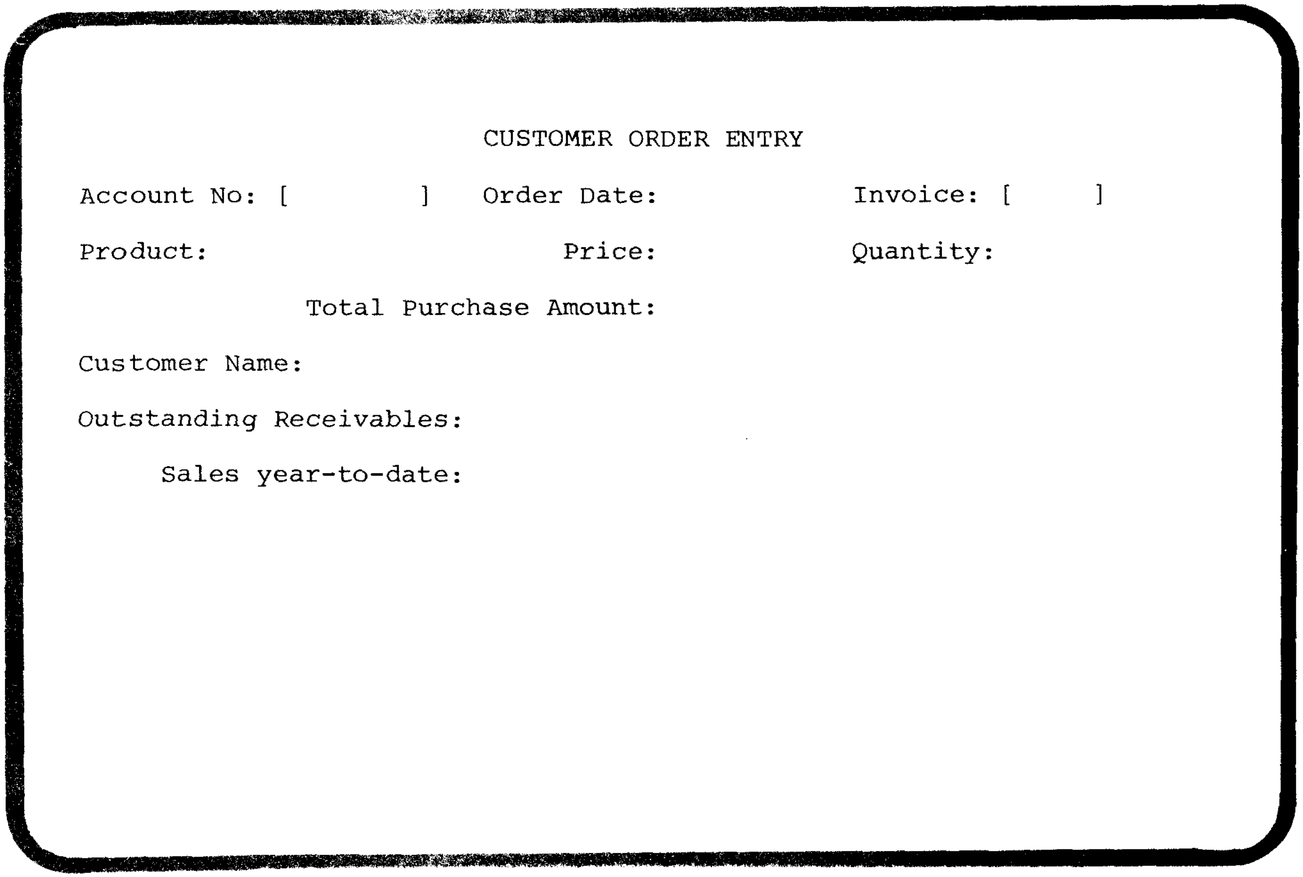
As we stated earlier, the Data Dictionary may be used to define or redefine any fields you want without having to go through the Screen Painting Editor. Remember, though, for the SENSIBLE SOLUTION to be aware of any changes, you must save the new definition with the **[ESC] [S]** (Save record) keys. When you finish changing definitions in the Data Dictionary, you must also re-initialize every data file (filename.MS/.KS) effected by the change and then re-compile the source code.

Now that we have created and initialized the new .MS/.KS data files for MEMORY and CUSTORD, it's time to paint the screen for our new sales order program.

You're an old hand at creating screens, so paint the screen shown on the opposite page. Specify CUSTORD as the file name and don't forget to enter the square brackets to help remind you which fields are key fields.

SCREEN FIELD WINDOW LABEL	**FIELD NAME**
Account No:	ORD.CUSCODE
Order date:	ORD.DATE
Invoice:	ORD.CUSINV
Product:	ORD.PROD
Price:	ORD.PRICE
Quantity:	ORD.QUANT

Paint this CUSTORD screen --



```
                         CUSTOMER ORDER ENTRY

Account No: [        ]   Order Date:           Invoice: [     ]

Product:                      Price:           Quantity:

              Total Purchase Amount:

Customer Name:

Outstanding Receivables:

     Sales year-to-date:
```

Total purchase amount:	ORD.TOTPURCHASE
Customer Name:	CUS.NAME
Outstanding receivables:	CUS.RECEIVE
Sales year to date:	CUS.SALES

When you are finished with your new art work save the screen and return to the Main Menu.

Use the Source Code Editor to create this program --

CUSTORD .SRR source file listing

```
          remark
          remark OPEN FILES:  CUSTFILE AND CUSTORD
BEGIN     trap RELATES gosub RELATES
          trap JUMP SCREEN goto CUSTFILE
          trap DELETE goto DELT.GRP
          mount screen CUSTORD
START     trap SAVE ignore
          enter ORD.CUSCODE
         find rec using field  ORD.CUSCODE  related field CUS.CUSCODE
                                                  on error goto CUSTFILE
          enter ORD.DATE
          enter ORD.CUSINV
          print at col 004  row 24 verbage PRODUCT CHOICES:  LANGUAGE,
                                                 MANAGEMENT, APPLICATIONS
          enter ORD.PROD
          print at col 004  row 24 verbage                              ]
          enter ORD.PRICE
          enter ORD.QUANT
          trap SAVE goto SAVE.GRP
          ORD.TOTPURCHASE = (ORD.PRICE)*(ORD.QUANT)
          CUS.RECEIVE = (CUS.RECEIVE)+(ORD.TOTPURCHASE)
          CUS.SALES = (CUS.SALES)+(ORD.TOTPURCHASE)
          N.2.0.1 = portion of ORD.DATE from 001 for 002 chars
          CUS.MONTH01 = &(N.2.0.1)(ORD.TOTPURCHASE)+(N.2.0.1)&(CUS.MONTH01)
SAVE.GRP  save rec in file CUSTORD confirm  / clear buffer if no save
                                                        then goto BEGIN
          save rec in file CUSTFILE no confirm / clear buffer
          goto START
DELT.GRP  delete rec in file CUSTORD confirm if no delete then goto BEGIN
          goto START
RELATES    find rec using field ORD.CUSCODE  related field CUS.CUSCODE
                                                  on error goto CUSTFILE
          return
CUSTFILE  trap JUMP SCREEN goto BEGIN
```

LESSON 7
Creating Our Sales-Order Program

Select "Main Menu 4) Source Code Editor" from the main menu. Using the Source Code Editor we want to create the CUSTORD.SRR file. Create the program listed on the opposite page.

Line 2 remark OPEN FILES: CUSTFILE AND CUSTORD

This line is for information purposes only.

Line 3 BEGIN   trap RELATES gosub RELATES

This command sets a trap for any kind of search and sends program control to the command line labeled RELATES.

Line 4 trap JUMP SCREEN goto CUSTFILE

This command sets a trap for the Jump Screen option and sends program control to the label CUSTFILE.

Line 5 trap DELETE goto DELT.GRP

This command sets a trap for the Remove record option and sends program control to the label DELT.GRP.

Line 6 mount screen CUSTORD

This command mounts the screen format on the terminal display.

Line 7 START   trap SAVE ignore

This command sets a trap for the Save record option specifying that it be ignored.

Line 8 enter ORD.CUSCODE

This command allows program user to enter the customer number.

Use the Source Code Editor to create this program --

CUSTORD .SRR source file listing

```
           remark
           remark OPEN FILES:  CUSTFILE AND CUSTORD
BEGIN      trap RELATES gosub RELATES
           trap JUMP SCREEN goto CUSTFILE
           trap DELETE goto DELT.GRP
           mount screen CUSTORD
START      trap SAVE ignore
           enter ORD.CUSCODE
           find rec using field  ORD.CUSCODE  related field CUS.CUSCODE
                                              on error goto CUSTFILE
           enter ORD.DATE
           enter ORD.CUSINV
           print at col 004  row 24 verbage PRODUCT CHOICES:  LANGUAGE,
                                           MANAGEMENT, APPLICATIONS
           enter ORD.PROD
           print at col 004  row 24 verbage                          ]
           enter ORD.PRICE
           enter ORD.QUANT
           trap SAVE goto SAVE.GRP
           ORD.TOTPURCHASE = (ORD.PRICE)*(ORD.QUANT)
           CUS.RECEIVE = (CUS.RECEIVE)+(ORD.TOTPURCHASE)
           CUS.SALES = (CUS.SALES)+(ORD.TOTPURCHASE)
           N.2.0.1 = portion of ORD.DATE from 001 for 002 chars
           CUS.MONTH01 = &(N.2.0.1)(ORD.TOTPURCHASE)+(N.2.0.1)&(CUS.MONTH01)
SAVE.GRP   save rec in file CUSTORD confirm  / clear buffer if no save
                                                      then goto BEGIN
           save rec in file CUSTFILE no confirm / clear buffer
           goto START
DELT.GRP   delete rec in file CUSTORD confirm if no delete then goto BEGIN
           goto START
RELATES    find rec using field ORD.CUSCODE  related field CUS.CUSCODE
                                              on error goto CUSTFILE
           return
CUSTFILE   trap JUMP SCREEN goto BEGIN
```

Line 9 find rec using field ORD.CUSCODE related field CUS.CUSCODE on error goto CUSTFILE

This command takes the entered customer number in ORD.CUSCODE and finds the related CUS.CUSCODE customer number record. If the record can not be located and an error occurs, program control jumps to the label CUSTFILE.

Line 10 enter ORD.DATE

This command allows program user to enter the date.

Line 11 enter ORD.CUSINV

This command allows program user to enter the invoice number.

Line 12 print at col 004 row 24 message PRODUCT CHOICES: LANGUAGE, MANAGEMENT, APPLICATIONS

This command displays the message at the bottom of the screen.

Line 13 enter ORD.PROD

This command allows program user to enter the product description.

Line 14 print at col 004 row 24   ]

(Place a right bracket at the end of the space provided for message.)

This command clears the previously displayed message.

Line 15 enter ORD.PRICE

This command allows program user to enter the product price.

Line 16 enter ORD.QUANT

This command allows program user to enter the amount of product that was ordered.

Use the Source Code Editor to create this program --

CUSTORD .SRR source file listing

```
           remark
           remark OPEN FILES:  CUSTFILE AND CUSTORD
BEGIN      trap RELATES gosub RELATES
           trap JUMP SCREEN goto CUSTFILE
           trap DELETE goto DELT.GRP
           mount screen CUSTORD
START      trap SAVE ignore
           enter ORD.CUSCODE
           find rec using field  ORD.CUSCODE  related field CUS.CUSCODE
                                                  on error goto CUSTFILE
           enter ORD.DATE
           enter ORD.CUSINV
           print at col 004  row 24 verbage PRODUCT CHOICES:  LANGUAGE,
                                                MANAGEMENT, APPLICATIONS
           enter ORD.PROD
           print at col 004  row 24 verbage                            ]
           enter ORD.PRICE
           enter ORD.QUANT
           trap SAVE goto SAVE.GRP
           ORD.TOTPURCHASE = (ORD.PRICE)*(ORD.QUANT)
           CUS.RECEIVE = (CUS.RECEIVE)+(ORD.TOTPURCHASE)
           CUS.SALES = (CUS.SALES)+(ORD.TOTPURCHASE)
           N.2.0.1 = portion of ORD.DATE from 001 for 002 chars
           CUS.MONTH01 = &(N.2.0.1)(ORD.TOTPURCHASE)+(N.2.0.1)&(CUS.MONTHO1)
SAVE.GRP   save rec in file CUSTORD confirm  / clear buffer if no save
                                                         then goto BEGIN
           save rec in file CUSTFILE no confirm / clear buffer
           goto START
DELT.GRP   delete rec in file CUSTORD confirm if no delete then goto BEGIN
           goto START
RELATES    find rec using field ORD.CUSCODE  related field CUS.CUSCODE
                                                  on error goto CUSTFILE
           return
CUSTFILE   trap JUMP SCREEN goto BEGIN
```

Line 17 trap SAVE goto SAVE.GRP

This command sets a trap for the Save record option and sends program control to the command line labeled SAVE.GRP.

Note: When writing the following "= Expression" commands, do not enter spaces in the expression line (eg. (CUS.SALES)+(ORD.TOTPURCHASE) is correct, while (CUS.SALES) + (ORD.TOTPURCHASE) is not). Your program will not compile correctly if you fail to observe this rule.

Line 18 ORD.TOTPURCHASE = (ORD.PRICE)*(ORD.QUANT)

Using the entered data, this expression will compute the total amount of purchase.

Line 19 CUS.RECEIVE = (CUS.RECEIVE)+(ORD.TOTPURCHASE)

This command updates the customer master outstanding receivables amount.

Line 20 CUS.SALES = (CUS.SALES)+(ORD.TOTPURCHASE)

This command updates the customer master year-to-date sales amount.

Line 21 N.2.0.1 = portion of ORD.DATE from 001 for 002 characters

This command "strips off" the first two characters of our date field, which the SENSIBLE SOLUTION considers to be "numeric" for calculation purposes, and assigns the value to our temporary Memory field N.2.0.1.

Line 22 CUS.MONTH01 = &(N.2.0.1)(ORD.TOTPURCHASE)+(N.2.0.1)&(CUS.MONTH01)

The expression line in this command is an array. Arrays are really just data tables. If you want to determine the value of a particular datum in a table, you simply find the appropriate table and row and then read the value. With the SENSIBLE SOLUTION the process is very similar; arrays provide a method for determining the location of the particular datum you want.

The SENSIBLE SOLUTION uses the "ampersand" character, & , to indicate a **table counter.** The value of the field associated with the ampersand (you **must** use a **variable** not a constant) tells the command file **which** element of a table to access.

Use the Source Code Editor to create this program --

CUSTORD .SRR source file listing

```
           remark
           remark OPEN FILES:  CUSTFILE AND CUSTORD
BEGIN      trap RELATES gosub RELATES
           trap JUMP SCREEN goto CUSTFILE
           trap DELETE goto DELT.GRP
           mount screen CUSTORD
START      trap SAVE ignore
           enter ORD.CUSCODE
          find rec using field  ORD.CUSCODE  related field CUS.CUSCODE
                                                 on error goto CUSTFILE
           enter ORD.DATE
           enter ORD.CUSINV
           print at col 004  row 24 verbage PRODUCT CHOICES:  LANGUAGE,
                                             MANAGEMENT, APPLICATIONS
           enter ORD.PROD
           print at col 004  row 24 verbage                          ]
           enter ORD.PRICE
           enter ORD.QUANT
           trap SAVE goto SAVE.GRP
           ORD.TOTPURCHASE = (ORD.PRICE)*(ORD.QUANT)
           CUS.RECEIVE = (CUS.RECEIVE)+(ORD.TOTPURCHASE)
           CUS.SALES = (CUS.SALES)+(ORD.TOTPURCHASE)
           N.2.0.1 = portion of ORD.DATE from 001 for 002 chars
           CUS.MONTH01 = &(N.2.0.1)(ORD.TOTPURCHASE)+(N.2.0.1)&(CUS.MONTH01)
SAVE.GRP   save rec in file CUSTORD confirm  / clear buffer if no save
                                                       then goto BEGIN
           save rec in file CUSTFILE no confirm / clear buffer
           goto START
DELT.GRP   delete rec in file CUSTORD confirm if no delete then goto BEGIN
           goto START
RELATES     find rec using field ORD.CUSCODE  related field CUS.CUSCODE
                                                 on error goto CUSTFILE
           return
CUSTFILE   trap JUMP SCREEN goto BEGIN
```

This allows you either to "read" an existing value or "write" a new value into a table.

Whether the & indicates a "read" or a "write" depends on where it appears in the calc expression. When the **&** appears **left** of the **table counter** (first character in the expression), it means "Write the calculated value in the (N)th position in the table starting at the specified field name."

&(value){calc expression}

assigns {calc expression} to the (value)th table entry in a table that **begins** with the specified field.

When the & is **part** of a calc expression, it appears between a numeric value (either a field **or** a constant) and the field name that begins the table...

+(value)&(field.NAME)

...and indicates that the (value)th table entry should be used in making the calculation.

We set up a table of sales-by-month when we created the Customer Master File, CUSTFILE, containing the fields **CUS.MONTH01**, **CUS.MONTH02** and so on. The SENSIBLE SOLUTION organizes fields within a file **in alphabetical order**, and as a result of the names we gave them, these twelve fields follow right after one another in the record. When creating a table in the SENSIBLE SOLUTION, you **must** give the table's field names in alphabetical order! If another field appears "in the middle", the SENSIBLE SOLUTION won't know how to access the data correctly.

If the date on the invoice is January, N.2.0.2=1 and the expression calculates the sum of "total purchase amount" (ORD.TOTPURCHASE) and the value in **CUS.MONTH01**. At this point, the above command assigns the result to **CUS.MONTH01**. If the invoice is dated July, N.2.0.1= 7. Total purchase is added to the seventh table entry from **CUS.MONTH01**, which is **CUS.MONTH07**, and is then assigned to the seventh table entry starting at the specified assignment field. This turns out to be **CUS.MONTH07** again!

If you understand how these tables work, you'll realize that the **seventh** table entry starting at **CUS.MONTH01** is also the

Use the Source Code Editor to create this program --

CUSTORD .SRR source file listing

```
0001                remark
0002                remark OPEN FILES:  CUSTFILE AND CUSTORD
0003 BEGIN          trap RELATES gosub RELATES
0004                trap JUMP SCREEN goto CUSTFILE
0005                trap DELETE goto DELT.GRP
0006                mount screen CUSTORD
0007 START          trap SAVE ignore
0008                enter ORD.CUSCODE
0009               find rec using field  ORD.CUSCODE  related field CUS.CUSCODE
                                                      on error goto CUSTFILE
0010                enter ORD.DATE
0011                enter ORD.CUSINV
0012                print at col 004  row 24 verbage PRODUCT CHOICES:  LANGUAGE,
                                                    MANAGEMENT, APPLICATIONS
0013                enter ORD.PROD
0014                print at col 004  row 24 verbage                          ]
0015                enter ORD.PRICE
0016                enter ORD.QUANT
0017                trap SAVE goto SAVE.GRP
0018                ORD.TOTPURCHASE = (ORD.PRICE)*(ORD.QUANT)
0019                CUS.RECEIVE = (CUS.RECEIVE)+(ORD.TOTPURCHASE)
0020                CUS.SALES = (CUS.SALES)+(ORD.TOTPURCHASE)
0021                N.2.0.1 = portion of ORD.DATE from 001 for 002 chars
0022                CUS.MONTH01 = &(N.2.0.1)(ORD.TOTPURCHASE)+(N.2.0.1)&(CUS.MONTHO1)
0023 SAVE.GRP      save rec in file CUSTORD confirm  / clear buffer if no save
                                                           then goto BEGIN
0024                save rec in file CUSTFILE no confirm / clear buffer
0025                goto START
0026 DELT.GRP       delete rec in file CUSTORD confirm if no delete then goto BEGIN
0027                goto START
0028 RELATES        find rec using field ORD.CUSCODE  related field CUS.CUSCODE
                                                      on error goto CUSTFILE
0029                return
0030 CUSTFILE       trap JUMP SCREEN goto BEGIN
```

sixth table entry in the table starting at **CUS.MONTH02**, and so on. This freedom lets you work within small portions of a table when appropriate.

BE CAREFUL! You may know how big a table really is, but **The SENSIBLE SOLUTION** only knows about fields and their starting positions within a record. A table entry is found by looking at the size of the "target" field, then moving "down the record" by that number of characters an appropriate number of times. The SENSIBLE SOLUTION will happily give you the 13th element of the **CUS.MONTH##** table, which happens to be the first few bytes of **CUS.CITY**! That's also why the fields have to appear in the record right next to each other and why the field names must be in alphabetical order.

Line 23 SAVE.GRP save rec in file CUSTORD confirm / clear buffer if no save then goto BEGIN

This command saves the current record to CUSTORD if you answer Y to the Save question. If you answer N the program control goes to the command line labeled BEGIN.

Line 24 save rec in file CUSTFILE no confirm / clear buffer

This command saves the current record CUSTFILE without asking if you wish to or not, and clears the buffer.

Line 25 goto START

This command transfers control of the program back to the line labeled START.

Line 26 DELT.GRP delete rec in file CUSTORD confirm if no delete then goto BEGIN

This command saves the current record to the file CUSTORD if you answer Y to the Delete question. If you answer no, program control is transferred to the line labeled BEGIN.

Line 27 goto START

This command transfers control of the program back to the line labeled START.

Use the Source Code Editor to create this program --

CUSTORD .SRR source file listing

```
           remark
           remark OPEN FILES:  CUSTFILE AND CUSTORD
BEGIN      trap RELATES gosub RELATES
           trap JUMP SCREEN goto CUSTFILE
           trap DELETE goto DELT.GRP
           mount screen CUSTORD
START      trap SAVE ignore
           enter ORD.CUSCODE
          find rec using field  ORD.CUSCODE  related field CUS.CUSCODE
                                                on error goto CUSTFILE
           enter ORD.DATE
           enter ORD.CUSINV
           print at col 004  row 24 verbage PRODUCT CHOICES:  LANGUAGE,
                                              MANAGEMENT, APPLICATIONS
           enter ORD.PROD
           print at col 004  row 24 verbage                      ]
           enter ORD.PRICE
           enter ORD.QUANT
           trap SAVE goto SAVE.GRP
           ORD.TOTPURCHASE = (ORD.PRICE)*(ORD.QUANT)
           CUS.RECEIVE = (CUS.RECEIVE)+(ORD.TOTPURCHASE)
           CUS.SALES = (CUS.SALES)+(ORD.TOTPURCHASE)
           N.2.0.1 = portion of ORD.DATE from 001 for 002 chars
           CUS.MONTH01 = &(N.2.0.1)(ORD.TOTPURCHASE)+(N.2.0.1)&(CUS.MONTHO1)
SAVE.GRP   save rec in file CUSTORD confirm  / clear buffer if no save
                                                       then goto BEGIN
           save rec in file CUSTFILE no confirm / clear buffer
           goto START
DELT.GRP   delete rec in file CUSTORD confirm if no delete then goto BEGIN
           goto START
RELATES    find rec using field ORD.CUSCODE  related field CUS.CUSCODE
                                                on error goto CUSTFILE
           return
CUSTFILE   trap JUMP SCREEN goto BEGIN
```

Line 28 RELATES find rec using ORD.CUSCODE related field CUS.CUSCODE on error goto CUSTFILE

When a search is executed, program control is transferred to this line. The "ORD.CUSCODE" field (account no.) is related to the Customer Master File account number field "CUS.CUSCODE." The CUS.CUSCODE field is searched to find the related record in the Customer Master File that holds the same value as that in the ORD.CUSCODE field. If the number doesn't exist and an error occurs, program control is transferred to the command line labeled CUSTFILE.

Line 29 return

(Use the **Go** command to bring up the **R**eturn option.)

Program control returns to the line following the gosub command which had diverted program flow.

Line 30 CUSTFILE trap JUMP SCREEN goto BEGIN

If the Jump screen option is executed during the next section of the program, control will transfer back to the command line labeled BEGIN.

The program, as it stands now, accepts user entry of the customer account number (ORD.CUSCODE). The ORD.CUSCODE field is related to the Customer Master File account number field CUS.CUSCODE. If the program succeeds in finding the related record which holds the same value, it fetches the current information for the customer, displays it on the screen, and will allow a sales order entry.

But what happens if the search for a related record fails? We don't want to allow entry of a sales order, unless we have a customer master record for the customer placing that order. Yet, it's cumbersome to force our users to exit from CUSTORD, open CUSTFILE and create a master record, and then return to CUSTORD again to enter the customer's order. The solution is simple! We'll incorporate the CUSTFILE program into the CUSTORD program.

Let's **[ESC] [Q]** from CUSTORD and answer **NO** to the "Exit completely?" question. The screen will clear, and the Source Code editor will allow you to load CUSTFILE.

You are going to mark a block of command lines, write the block to a disk file, and then read the disk file into CUSTORD. Use the **[F] (Find line)** option to position the line prompt > at line 4 and **[M] (Mark block)**

CUSTORD.SRR as it appears after appending TEMPFILE --

```
0001                remark
0002                remark OPEN FILES:  CUSTFILE AND CUSTORD
0003 BEGIN          trap RELATES gosub RELATES
0004                trap JUMP SCREEN goto CUSTFILE
0005                trap DELETE goto DELT.GRP
0006                mount screen CUSTORD
0007 START          trap SAVE ignore
0008                enter ORD.CUSCODE
0009               find rec using field  ORD.CUSCODE  related field CUS.CUSCODE
                                                        on error goto CUSTFILE
0010                enter ORD.DATE
0011                enter ORD.CUSINV
0012                print at col 004  row 24 verbage PRODUCT CHOICES:  LANGUAGE,
                                                      MANAGEMENT, APPLICATIONS
0013                enter ORD.PROD
0014                print at col 004  row 24 verbage                          ]
0015                enter ORD.PRICE
0016                enter ORD.QUANT
0017                trap SAVE goto SAVE.GRP
0018                ORD.TOTPURCHASE = (ORD.PRICE)*(ORD.QUANT)
0019                CUS.RECEIVE = (CUS.RECEIVE)+(ORD.TOTPURCHASE)
0020                CUS.SALES = (CUS.SALES)+(ORD.TOTPURCHASE)
0021                N.2.0.1 = portion of ORD.DATE from 001 for 002 chars
0022                CUS.MONTH01 = &(N.2.0.1)(ORD.TOTPURCHASE)+(N.2.0.1)&(CUS.MONTH01)
0023 SAVE.GRP       save rec in file CUSTORD confirm  / clear buffer if no save
                                                               then goto BEGIN
0024                save rec in file CUSTFILE no confirm / clear buffer
0025                goto START
0026 DELT.GRP     delete rec in file CUSTORD confirm if no delete thengoto BEGIN
0027                goto START
0028 RELATES        find rec using field ORD.CUSCODE  related field CUS.CUSCODE
                                                        on error goto CUSTFILE
0029                return
0030 CUSTFILE       trap JUMP SCREEN goto BEGIN
0031                mount screen CUSTFILE
0032 START          enter CUS.CUSCODE
0033                if duplicate key using field CUS.CUSCODE goto START
0034                enter CUS.DATE
0035                enter CUS.NAME
0036                enter CUS.ADDR
0037                enter CUS.CITY
0038                enter CUS.STATE
0039                enter CUS.ZIP
```

to mark the beginning of the block. Using the same process, find line 12 and mark the end of the block.

Now select the **Write block [W]** option. The Source Code editor will request:

Enter the name of the file in which the block is to be saved: ?:******

Let's name our file **TEMPFILE.** We're finished with CUSTFILE so **[Q] Quit** and **[L] Load** CUSTORD. Position the line prompt > at the end of the program and select [R] **Read block.**

Enter the name of the file to be read: ?:******

Type **TEMPFILE.** The commands saved to the disk from your CUSTFILE program will be appended to the end of your CUSTORD program.

Now we have a program which allows the user to create new master customer records as necessary, using the CUSTFILE screen. Every time our program user enters a sales order for a customer who does not have a record in the master file, the program will immediately display the CUSTFILE screen. Let's print a message at the bottom of the screen display that offers options and directions to the program user.

Insert this command line following line 31:

print at col 000 row 00 message CUSTOMER NOT FOUND {RETURN} TO CREATE,{ESC} {ESC} JUMP TO REENTER

This command displays the message at the bottom of the screen. The program user is informed that there is no record for the customer account number just entered. If the user presses [RETURN] they can create the master file record at that time. If no related record could be found because the user entered an incorrect account number, they may [ESC] [ESC] [JUMP] to the CUSTORD screen to re-enter the number.

When our program user decides to create a master file record for a newly entered customer account number, it will help if their CUSTFILE screen displays the new account number which they entered to the CUSTORD screen. Insert the following command line after line 32:

CUS.CUSCODE = (ORD.CUSCODE)

Let's continue by examining the command lines we just appended to our program.

Line 31 mount screen CUSTFILE

CUSTORD.SRR as it appears after altering lines 32 and 33 --

```
0001           remark
0002           remark OPEN FILES:  CUSTFILE AND CUSTORD
0003 BEGIN     trap RELATES gosub RELATES
0004           trap JUMP SCREEN goto CUSTFILE
0005           trap DELETE goto DELT.GRP
0006           mount screen CUSTORD
0007 START     trap SAVE ignore
0008           enter ORD.CUSCODE
0009          find rec using field  ORD.CUSCODE  related field CUS.CUSCODE
                                              on error goto CUSTFILE
0010           enter ORD.DATE
0011           enter ORD.CUSINV
0012           print at col 004  row 24 verbage PRODUCT CHOICES:  LANGUAGE,
                                           MANAGEMENT, APPLICATIONS
0013           enter ORD.PROD
0014           print at col 004  row 24 verbage                            ]
0015           enter ORD.PRICE
0016           enter ORD.QUANT
0017           trap SAVE goto SAVE.GRP
0018           ORD.TOTPURCHASE = (ORD.PRICE)*(ORD.QUANT)
0019           CUS.RECEIVE = (CUS.RECEIVE)+(ORD.TOTPURCHASE)
0020           CUS.SALES = (CUS.SALES)+(ORD.TOTPURCHASE)
0021           N.2.0.1 = portion of ORD.DATE from 001 for 002 chars
0022           CUS.MONTH01 = &(N.2.0.1)(ORD.TOTPURCHASE)+(N.2.0.1)&(CUS.MONTH01)
0023 SAVE.GRP  save rec in file CUSTORD confirm  / clear buffer if no save
                                                        then goto BEGIN
0024           save rec in file CUSTFILE no confirm / clear buffer
0025           goto START
0026 DELT.GRP delete rec in file CUSTORD confirm if no delete thengoto BEGIN
0027           goto START
0028 RELATES   find rec using field ORD.CUSCODE  related field CUS.CUSCODE
                                              on error goto CUSTFILE
0029           return
0030 CUSTFILE  trap JUMP SCREEN goto BEGIN
0031           mount screen CUSTFILE
0032          print  at col 000 row 00 verbage CUSTOMER NOT FOUND {RETURN}
                                  TO CREATE,{ESC} {ESC} JUMP TO REENTER
0033           CUS.CUSCODE = (ORD.CUSCODE)
0034           enter CUS.DATE
0035           enter CUS.NAME
0036           enter CUS.ADDR
0037           enter CUS.CITY
0038           enter CUS.STATE
0039           enter CUS.ZIP
```

This command mounts the screen CUSTFILE on the terminal display.

Line 32 print at col 000 row 00 message CUSTOMER NOT FOUND {RETURN} TO CREATE,{ESC} {ESC} JUMP TO REENTER

This command displays the message at the bottom of the screen.

Line 33 CUS.CUSCODE =(ORD.CUSCODE)

This command specifies that the CUS.CUSCODE field will display the value which resides in the ORD.CUSCODE field. In other words, CUS.CUSCODE will become equal to ORD.CUSCODE.

The next command line was labeled START in the CUSTFILE program. Use the "Delete line option" to eliminate this line. Delete as well the next command line:

if duplicate key using field CUS.CUSCODE goto START

These lines can be eliminated because CUS.CUSCODE has been assigned the value ORD.CUSCODE. ORD.CUSCODE's value has been tested in the earlier command line "find related record" (line 9).

Line 34 enter CUS.DATE

This command allows program user to enter the date.

Line 35 enter CUS.NAME

This command allows program user to enter the customer name.

Line 36 enter CUS.ADDR

This command allows program user to enter the customer address.

Line 37 enter CUS.CITY

This command allows program user to enter the city.

Line 38 enter CUS.STATE

This command allows program user to enter the state.

A **partial** listing of CUSTORD.SRR (beginning with line 6) as it appears at the end of our editing session --

```
0006            mount screen CUSTORD
0007 START      trap SAVE ignore
0008            enter ORD.CUSCODE
0009           find rec using field  ORD.CUSCODE  related field CUS.CUSCODE
                                                        on error goto CUSTFILE
0010            enter ORD.DATE
0011            enter ORD.CUSINV
0012            print at col 004  row 24 verbage PRODUCT CHOICES:  LANGUAGE,
                                                      MANAGEMENT, APPLICATIONS
0013            enter ORD.PROD
0014            print at col 004  row 24 verbage                             ]
0015            enter ORD.PRICE
0016            enter ORD.QUANT
0017            trap SAVE goto SAVE.GRP
0018            ORD.TOTPURCHASE = (ORD.PRICE)*(ORD.QUANT)
0019            CUS.RECEIVE = (CUS.RECEIVE)+(ORD.TOTPURCHASE)
0020            CUS.SALES = (CUS.SALES)+(ORD.TOTPURCHASE)
0021            N.2.0.1 = portion of ORD.DATE from 001 for 002 chars
0022            CUS.MONTH = &(N.2.0.1)(ORD.TOTPURCHASE)+(N.2.0.1)&(CUS.MONTHO1)
0023 SAVE.GRP   save rec in file CUSTORD confirm  / clear buffer if no save
                                                               then goto BEGIN
0024            save rec in file CUSTFILE no confirm / clear buffer
0025            goto START
0026 DELT.GRP  delete rec in file CUSTORD confirm if no delete thengoto BEGIN
0027            goto START
0028 RELATES    find rec using field ORD.CUSCODE  related field CUS.CUSCODE
                                                        on error goto CUSTFILE
0029            return
0030 CUSTFILE   trap JUMP SCREEN goto BEGIN
0031            mount screen CUSTFILE
0032           print  at col 000 row 00 verbage CUSTOMER NOT FOUND {RETURN}
                                   TO CREATE,{ESC} {ESC} JUMP TO REENTER
0033            CUS.CUSCODE = (ORD.CUSCODE)
0034            enter CUS.DATE
0035            enter CUS.NAME
0036            enter CUS.ADDR
0037            enter CUS.CITY
0038            enter CUS.STATE
0039            enter CUS.ZIP
0040           save  rec in file CUSTFILE confirm/ clear buffer if no save
                                                               then goto BEGIN
0041            goto BEGIN
```

Line 39 enter CUS.ZIP

This command allows program user to enter the zipcode.

We have given our program users the means to create new customer master records. If they do so, we need a command in the program which saves this information and stores it in the CUSTFILE program. Insert the following commands:

Line 40 save rec in file CUSTFILE confirm / clear buffer if no save then goto BEGIN

This command saves the current record to CUSTFILE if you answer Y to the Save question and clears the buffer. If you answer N to the Save question, program control is transferred to the command line labeled BEGIN.

Line 41 goto BEGIN

This command transfers program control back to the command line labeled BEGIN.

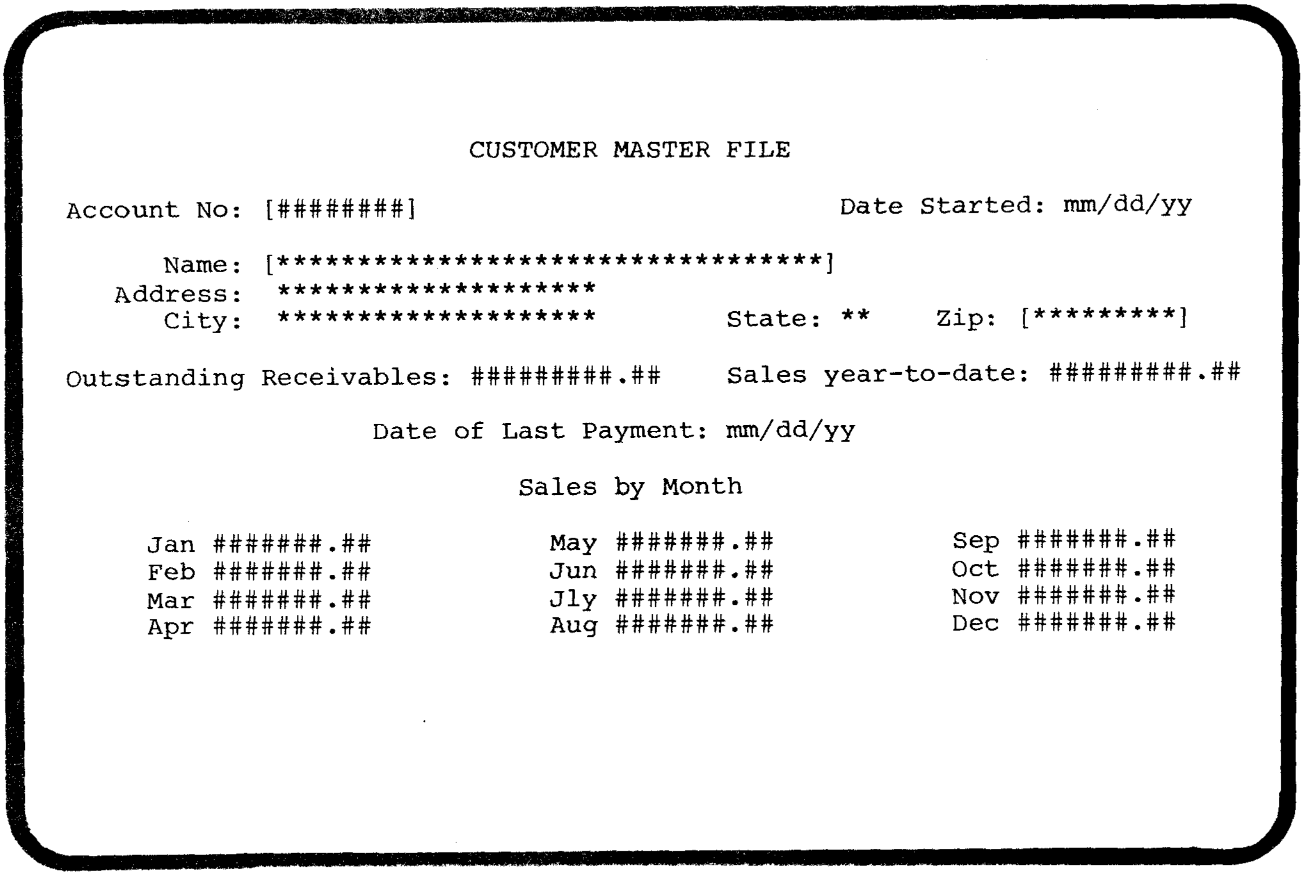
Now leave the Source Code Editor, and return to the Main Menu. Initialize CUSTORD with "Main Menu 5)Initialize a Data File. Finally, compile the CUSTORD program with "Main Menu 6) Compile Source Code."

Take a moment to look at the SENSIBLE SOLUTION Main Menu. Menu selections 2 through 6 present the logical progression of steps we have taken to create our CUSTORD program.

Selection 2-- Create files and fields in the Data Dictionary.
Selection 3-- Paint the program screen.
Selection 4-- Create the source code file.
Selection 5-- Initialize the data file.
Selection 6-- Compile the source code file.

Now try executing CUSTORD (Main Menu selection 1) and entering a few sales orders. You can scan and change the order file with the usual control keys. Use **[ESC]** [Q] to get back to the SENSIBLE SOLUTION main menu; then run CUSTPYMT to enter a few payments. When you later run CUSTFILE, you will see that the orders and payments have been posted to the correct fields in the Customer Master File.

Use **[ESC] [L] add Lne** to insert blank lines into your CUSTFILE screen, then put the "Date of last payment:" label on the screen and **ESC] [A] Add fld** CUS.LASTPYMT.



```
                         CUSTOMER MASTER FILE

Account No: [########]                          Date Started: mm/dd/yy

      Name: [***********************************]
   Address:  ********************
      City:  ********************       State: **    Zip: [*********]

Outstanding Receivables: #########.##   Sales year-to-date: #########.##

                   Date of Last Payment: mm/dd/yy

                            Sales by Month

     Jan #######.##            May #######.##           Sep #######.##
     Feb #######.##            Jun #######.##           Oct #######.##
     Mar #######.##            Jly #######.##           Nov #######.##
     Apr #######.##            Aug #######.##           Dec #######.##
```

LESSON 8
Adding New Fields to a File or Screen

One of the strongest features of the SENSIBLE SOLUTION is its ability to build on the work we've already done by adding new files, screens, and fields. Here's an example. When we call up a customer's master record we'd like to know when the customer made his last payment. That information is entered on the payment screen. We need to modify the **CUSTFILE** screen to show this date, change the Data Dictionary entries for the **CUSTFILE** data file to add a field to hold this date, and also modify **CUSTPYMT** to post the information over to the **CUSTFILE.MS** data file.

Since we're adding a new field to the file, we must also **restructure** the record layout to make room for the new data. The SENSIBLE SOLUTION provides an option from the main menu to do this automatically for us, "Main Menu 8) Restructure A Data File." Once we've modified our two program/screens, we'll be using this program.

From the SENSIBLE SOLUTION main menu, choose 3) SCREEN PAINTING and modify **CUSTFILE.** Put the "Date of last payment:" label on the screen, and add this field just to the right:

Date of last payment:	Field Name:	CUS.LASTPYMT
	File Name:	CUSTFILE
	Field Type:	D
	Key:	N

Then **[ESCAPE]** [Q] to exit from the Screen Painting editor and save the screen.

When the SENSIBLE SOLUTION main menu returns, select 8) Restructure A Data File. The following prompt will appear on your screen:

Enter the name of the file to be restructured ******

Enter the name of the data file CUSTFILE. The following message will be displayed:

Create work file CUSTFILE.M$$ on drive _:

RESTRUCTURE has to make a work file before it creates the new restructured data file. Thus, for a time, your disk storage will have to contain an amount of data that is twice the size of the original data file. To provide

SENSIBLE SOLUTION File Extensions --

File extensions created and used with SENSIBLE SOLUTION:

*.SCC	Screen or Reporter Format files
*.SRR	Command Source Code files
*.RUN	Compiled Command files
*.IQ	Inquire Format files
*.LST	Screen/Reporter Format or Source Code ASCII files
*.MS	Master Data file
*.KS	Key file (for the .MS file)

Operating System file extensions used with SENSIBLE SOLUTION:

*.COM	Compiled executible program - CP/M, MP/M
*.EXE	Compiled executible program - MS DOS, PC DOS

for this extra work space, RESTRUCTURE will allow you to designate a different drive location to contain the temporary work file. For example, if your original data file is 1000K bytes in size, you will need at least 2000K bytes of free space on this same drive to avoid the necessity of designating a unique temporary work drive location. If your present drive location does not have enough free space, type in a different drive location letter than the one displayed.

Press **[RETURN]** for the default drive, or give the letter of any drive with enough room left on the disk for the temporary file. Restructure will abort if the work drive has too little free space.

After entering a valid file name and drive location, RESTRUCTURE will immediately begin execution. Sit back and watch the program run. You will see SENSIBLE SOLUTION go through building the restructure table, restructuring the records, re-writing the Data Dictionary, and replacing the old file. When the program is finished, a message will appear to remind you to REKEY the file and RECOMPILE all programs that access the data file:

Restructure is complete. Please remember to
--REKEY your file, and
--RECOMPILE all programs which use this file

Suppose you had added or deleted a key field, altered a key field's size, or redefined a key field as a non-key field, within your file. Any time that a different set of key fields has been defined in an existing data file, the index-key file, **filename.KS**, must be rebuilt and the record counter in the file must be reset to reflect the total number of records in the file. Main Menu option 7) Rekey a Data File performs these functions by re-keying the SENSIBLE SOLUTION data file, filename.KS. In fact, we recommend that you use option 7 any time that an existing data file containing "live" data has been restructured.

Go ahead and REKEY now. Enter the file name **CUSTFILE.** To the prompt "Number of records to be re-keyed" respond with **[RETURN]** to indicate that you wish to rekey the entire file. Then watch the REKEY program work on each record in the file.

The **CUSTFILE**, **CUSTORD** and **CUSTPYMT** programs all use the newly restructured CUSTFILE data file. To accommodate the new file structure, you **must** recompile these files. Recompiling will alert SENSIBLE SOLUTION to the changes made in the CUSTFILE.SCC screen format and update the .SRR and .RUN files. Recompile CUSTFILE AND CUSTORD now, using "Main Menu 6)

CUSTPYMT.SRR as it appears after inserting line 9 --

CUSTPYMT.SRR source file listing

```
0001            remark
0002            remark OPEN FILES:  CUSTFILE AND CUSTPYMT
0003            trap SAVE goto SAVE.GRP
0004            trap DELETE goto DELT.GRP
0005            mount screen CUSTPYMT
0006 START      enter PYMT.CUSCODE
0007            find rec using field PYMT.CUSCODE  related field CUS.CUSCODE
                                                        on error goto START
0008            enter PYMT.DATE
0009            CUS.LASTPYMT = (PYMT.DATE)
0010            enter PYMT.PAYMENT
0011            CUS.RECEIVE = (CUS.RECEIVE)-(PYMT.PAYMENT)
0012 SAVE.GRP   save rec in file CUSTPYMT confirm / clear buffer
0013            save rec in file CUSTFILE no confirm / clear buffer
0014            goto START
0015 DELT.GRP   delete rec in file CUSTPYMT confirm
0016            goto START
```

Compile a Source Code File."

Since the **CUSTFILE** program/screen doesn't accept data entry to the "Last Payment Date" field, you don't have to modify that program; however, you **do** have to change the CUSTPYMT command source code file **CUSTPYMT.SRR** to post the date over to the **CUSTFILE.MS** data file.

From The SENSIBLE SOLUTION main menu, select "4) Source Code Editor" to modify CUSTPYMT. Insert the following command after line 8:

```
CUS.LASTPYMT = (PYMT.DATE)
```

This will place the payment date in the customer record (overwriting the previous "last payment" entry) every time a payment is made.

Because you have modified the CUSTPYMT source code file, and the CUSTFILE screen format file which it accesses, it is necessary to re-compile CUSTPYMT to update SENSIBLE SOLUTION's CUSTPYMT.RUN file. Do so now, using "Main Menu 6) Compile Source Code."

Use the Source Code Editor to create the following program --

CUSMENU .SRR source file listing

```
0001            remark
0002            remark OPEN FILES:  CUSMENU
0003            mount screen CUSMENU
0004 START      enter N.1.0.1
0005            goto line on value of N.1.0.1 maximum gotos 03 if error goto
                                                                          START
0006            execute .Run file CUSTFILE
0007            execute .Run file CUSTPYMT
0008            execute .Run file CUSTORD
```

LESSON 9
Creating a Menu

```
OUR COMPANY ACCOUNTING SYSTEM
-----------------------------

 1) Maintain Customer File
 2) Record Customer Payments
 3) Record Customer Orders

#  What is your selection?
```

It's becoming tedious to return to the SENSIBLE SOLUTION main menu every time and use 1) Execute A SENSIBLE SOLUTION Program to run our programs. Let's create a **menu** screen that will allow us to execute our programs directly from the menu.

Actually, the SENSIBLE SOLUTION main menu is **itself** a SENSIBLE SOLUTION program called **MENU.RUN.** You can call it up and list it via "Main Menu 4) Source Code Editor" to see how a menu program is constructed. **DON'T EDIT THE MAIN MENU**, for heaven's sake!!!

Create a screen called **CUSMENU** to match the one shown above . There is only one field to add. Position your cursor over the "#" sign and **ESC] [A]**:

Selection	Field name:	N.1.0.1
	File name:	MEMORY
	Field type:	N
	Size:	1
	Number of decimal places:	0
	Key:	N

Save this screen. Then, since you added a field to the file, **MEMORY**, you must **initialize** it. Select "Main Menu 5) Initialize a Data File" and enter the file name **MEMORY.**

Now use "Main Menu 4) Source Code Editor" to create the source code file (CUSMENU.SRR) shown on the facing page and **compile** that listing.

In this source listing there are two commands we haven't encountered yet. **GOTO LINE ON VALUE** tests the value of the specified field, which should be a

CUSTFILE after modification --

CUSTFILE.SRR source file listing

```
           remark
           trap EXIT goto EXIT
           trap SAVE goto SAVE.GRP
           trap DELETE goto DELT.GRP
           mount screen CUSTFILE
START      enter CUS.CUSCODE
           enter CUS.DATE
           if duplicate key using field CUS.CUSCODE goto START
           enter CUS.NAME
           enter CUS.ADDR
           enter CUS.CITY
           enter CUS.STATE
           enter CUS.ZIP
SAVE.GRP   save rec in file CUSTFILE confirm / clear buffer
           goto START
DELT.GRP   delete rec in file CUSTFILE confirm
           goto START
EXIT       execute .Run file CUSMENU
```

numeric type. If the value is less than one or greater than the **maximum** value, control of the program is transferred to the **goto** target label -- **START.** Otherwise, control "jumps" to one of the statements that follow the GOTO LINE ON VALUE depending on the value of the field, **N.1.0.1.** If **N.1.0.1** is one, the next command to be executed is the **first** command following the GOTO LINE ON VALUE, which is line 6. If **N.1.0.1** = 2, line 7 is executed. **N.1.0.1** = 3 triggers line 8. **N.1.0.1** = 4 is greater than the **maximum** so the program would "goto" START instead of to one of the commands following the GOTO LINE ON VALUE.

GOTO LINE ON VALUE's action is sometimes referred to as a "table jump". Control is passed to the **n**th element of a **table** of commands, depending on the value of **n.** It is used often in menus like this. It might also be used to perform several different tasks depending, say, on the month of a data entry.

EXECUTE .RUN FILE, executes another .RUN file. In this example, if you entered **1** for your selection choice, the CUSTFILE screen will appear and allow you to update master customer records.

Now we need to modify **CUSTFILE, CUSTPYMT** and **CUSTORD** so that they can transfer control between screens correctly and return to the **CUSMENU** screen on **[ESC] [Q].** You'll find the command lines to insert listed below. Use "Main Menu 4) Source Code Editor" to enter the required modifications and then "Main Menu 6) Compile Source Code" on each file to make a "runable" program from the modified command file.

Insert this command line at the beginning of each one of your three programs, following the "remark" line:

```
trap EXIT goto EXIT
```

Now use the End option to reach the end of your programs and then the Insert mode to insert the following line:

```
EXIT          execute .Run file CUSMENU
```

The new **TRAP EXIT** command is activated whenever the program user presses the **[ESC] [Q]** keys.

In these files, "exiting" the screen via **[ESC] [Q]** will transfer control to the new line labeled EXIT which, in turn, runs our **CUSMENU.RUN** file.

CUSTPYMT after modification --

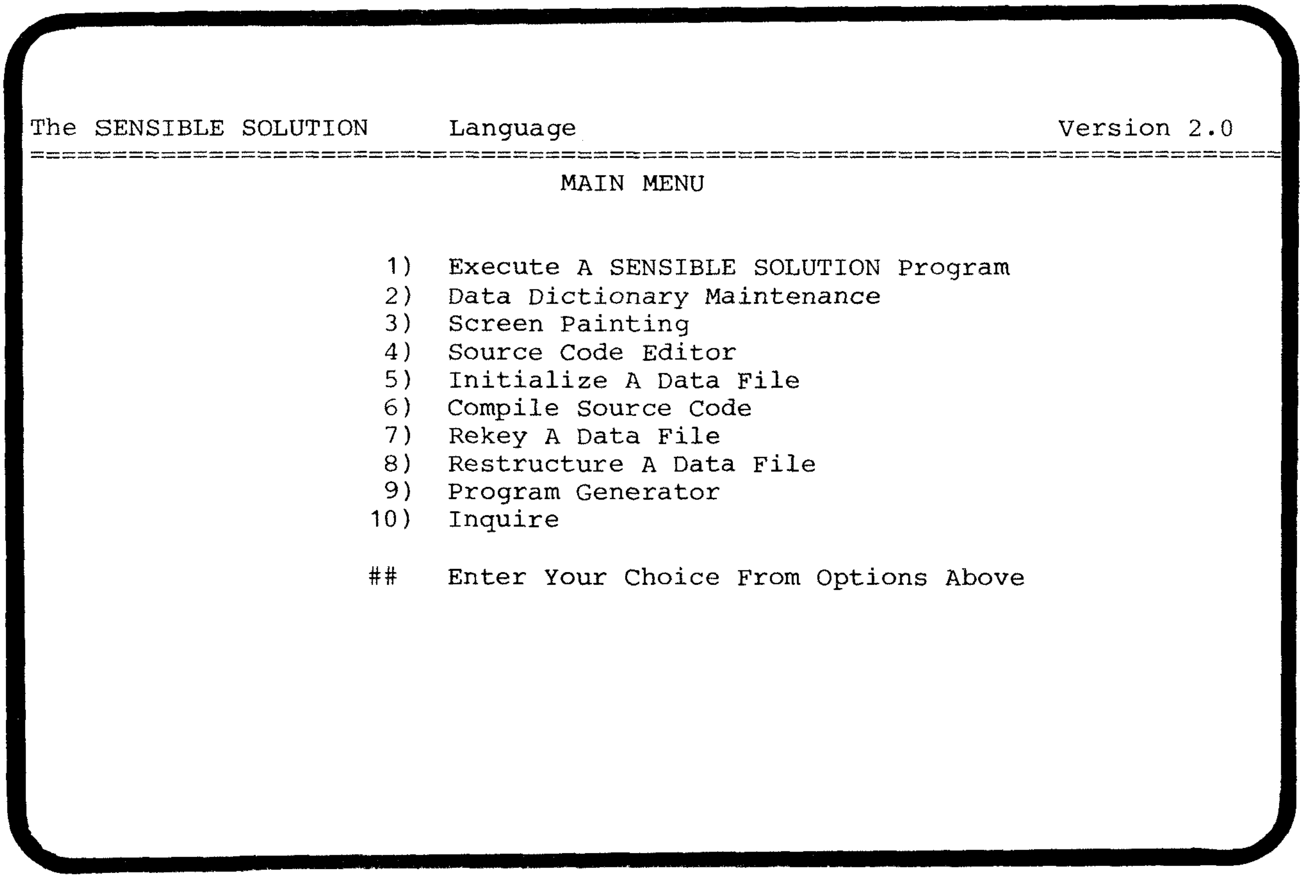
CUSTPYMT.SRR source file listing

```
           remark
           remark OPEN FILES:  CUSTFILE AND CUSTPYMT
           trap EXIT goto EXIT
           trap SAVE goto SAVE.GRP
           trap DELETE goto DELT.GRP
           mount screen CUSTPYMT
START      enter PYMT.CUSCODE
           find rec using field PYMT.CUSCODE  related field CUS.CUSCODE
                                                    on error goto START
           enter PYMT.DATE
           CUS.LASTPYMT = (PYMT.DATE)
           enter PYMT.PAYMENT
           CUS.RECEIVE = (CUS.RECEIVE)-(PYMT.PAYMENT)
SAVE.GRP   save rec in file CUSTPYMT confirm / clear buffer
           save rec in file CUSTFILE no confirm / clear buffer
           goto START
DELT.GRP   delete rec in file CUSTPYMT confirm
           goto START
EXIT       execute .Run file CUSMENU
```

The programs now completed can form the backbone of a customer order-entry/payment-control system and can be integrated into a complete accounting system. Of course, you'll want to make some additions to the present programs. For instance, the way our system is currently designed, you can change the amount of a payment, which would also change the receivables and sales figures, with no record that a change was made. (Someone get a bucket of water. All the accountants just fainted!)

Before you graduate as a SENSIBLE SOLUTION applications programmer, we'd like to introduce you to two more features of the SENSIBLE SOLUTION: "Main Menu 10) Inquire", a fast and easy utility for making quick "ad-hoc" queries of SENSIBLE SOLUTION data files, and "Main Menu 3) Screen Painting " -- Reporter Format, a powerful and flexible report-generating facility.

After typing SENSIBLE **[RETURN]** --



```
The SENSIBLE SOLUTION     Language                                Version 2.0
==============================================================================
                                MAIN MENU


                     1)   Execute A SENSIBLE SOLUTION Program
                     2)   Data Dictionary Maintenance
                     3)   Screen Painting
                     4)   Source Code Editor
                     5)   Initialize A Data File
                     6)   Compile Source Code
                     7)   Rekey A Data File
                     8)   Restructure A Data File
                     9)   Program Generator
                    10)   Inquire

                    ##    Enter Your Choice From Options Above
```

LESSON 10
Inquire, the Quick Data Scanning Facility

If you use a computer application every day, you quickly build up a valuable base of information -- a "data base". Unfortunately, with most languages, there is no way to get at all that information.

As you have learned, with SENSIBLE SOLUTION it's a simple matter to design a program to answer any question you have. And, for questions that you ask over and over again, that's exactly what you should do. However, it often happens that you need a quick answer to a simple question. For example, "What products have we sold to our customers this year?"

The SENSIBLE SOLUTION provides a facility to answer just this sort of question -- Inquire. Inquire can scan any single SENSIBLE SOLUTION data file and create a report which can then be sent to the terminal, the printer, or to a disk file. With Inquire, you decide on the format of that report: what records will be selected from the file and how those records will be displayed on the report. You can then save the format and execute the Inquire report any time that you wish. The report will always display the most current data from your data base.

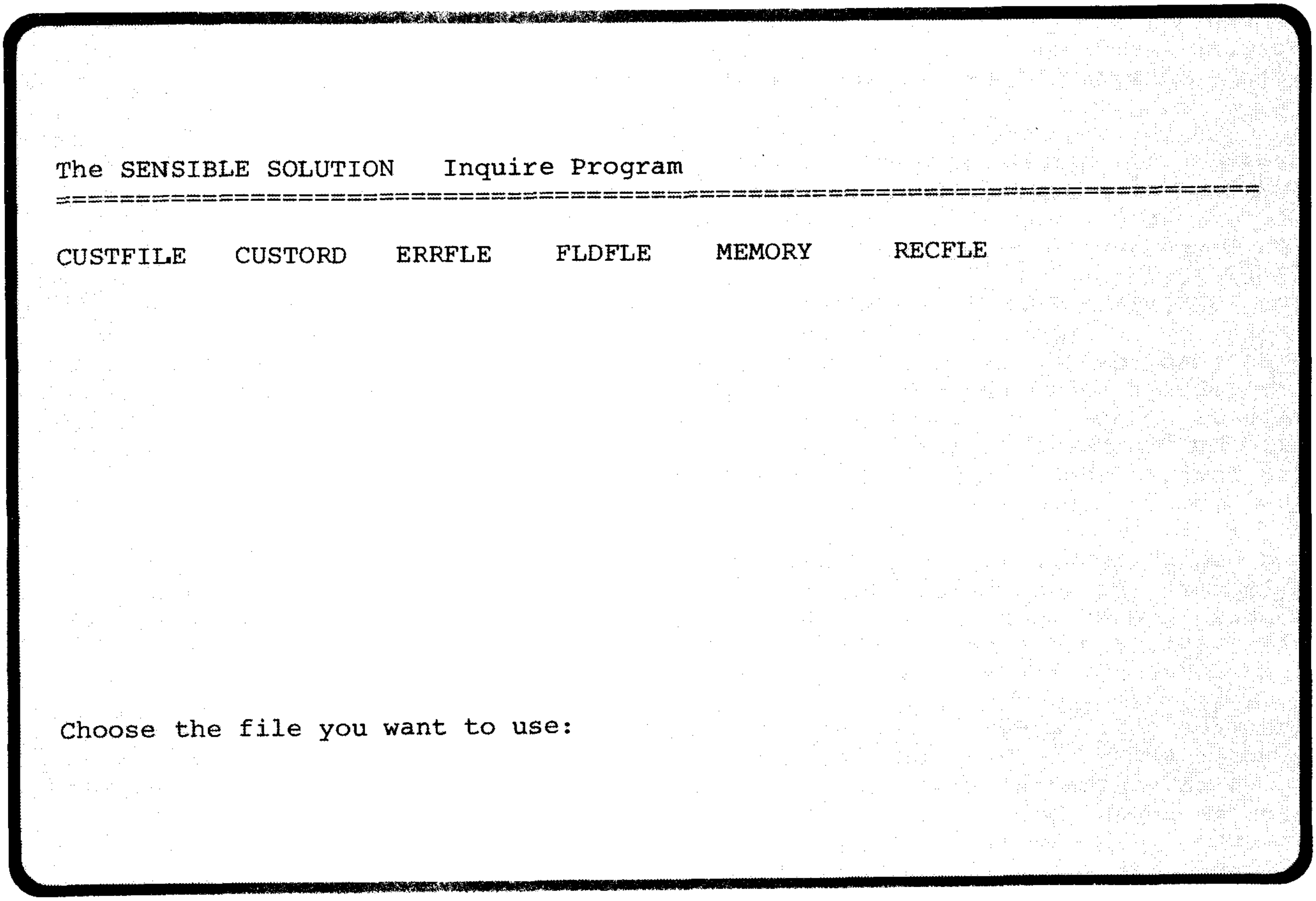
You begin using Inquire by selecting a file. Inquire will then display a directory of all the fields in that file and ask which of these fields you wish to see on your Inquire report. In this way, you need not display all the "sales-by-month" fields, for example, when you simply want "receivables", "name" and "telephone".

Inquire will create a report with the fields you specify in the order you specify them. Each field column on the report is automatically separated by a few spaces but you may, in fact, specify exactly how many spaces you want to appear between each column. Each record that is printed out on your report may be formatted on from 1 to 4 lines of the report. You can also insert blank lines in your report format.

Let's use Inquire to view some of the records in our data file, CUSTORD. We'll create a report format that will show account activity by listing customer account numbers, the date of the sales order, the product that was ordered, and the total amount payable. While we're at it, let's restrict the report to displaying only transactions made during 1984. With Inquire, this will all be easy!

Select "Main Menu 10) Inquire", and the screen shown on the next page will be displayed on your terminal. Inquire will begin by displaying a directory

"Main Menu 10) Inquire" --



```
The SENSIBLE SOLUTION   Inquire Program
===========================================================================

CUSTFILE   CUSTORD   ERRFLE    FLDFLE    MEMORY     RECFLE










Choose the file you want to use:
```

of all data files and then request that you enter a file name from which you wish to create a quick report. If you have too many data files to be displayed on your screen at once, use the **[ESC]** key to display a menu that will allow you to scroll your screen up and down to see the rest of the directory. Type in the file name CUSTORD and you will see the following prompt:

```
     Do you want Fieldname headings on report? (Y or N):
```

In a moment you will be selecting the fields that will be printed on the report in columnar form. Answering "Y" to this prompt will force Inquire to print a column heading that will appear at the top of every page of the report as it is printed out. For our CUSTORD report, go ahead and answer "Y".

The next prompt,

```
     Do you wish to substitute Fieldname headings? (Y or N):
```

will give you the option of either using the actual field names as headings on your report or substituting a unique heading that you can specify. If you answer "Y" to the above prompt, every time that you select a field to be printed on the report, another prompt will soon follow requesting that you specify whatever heading you want to see printed at the top of each page of the report. Inquire will ask for a field then Inquire will ask for a heading. Once again, answer "Y" to this question.

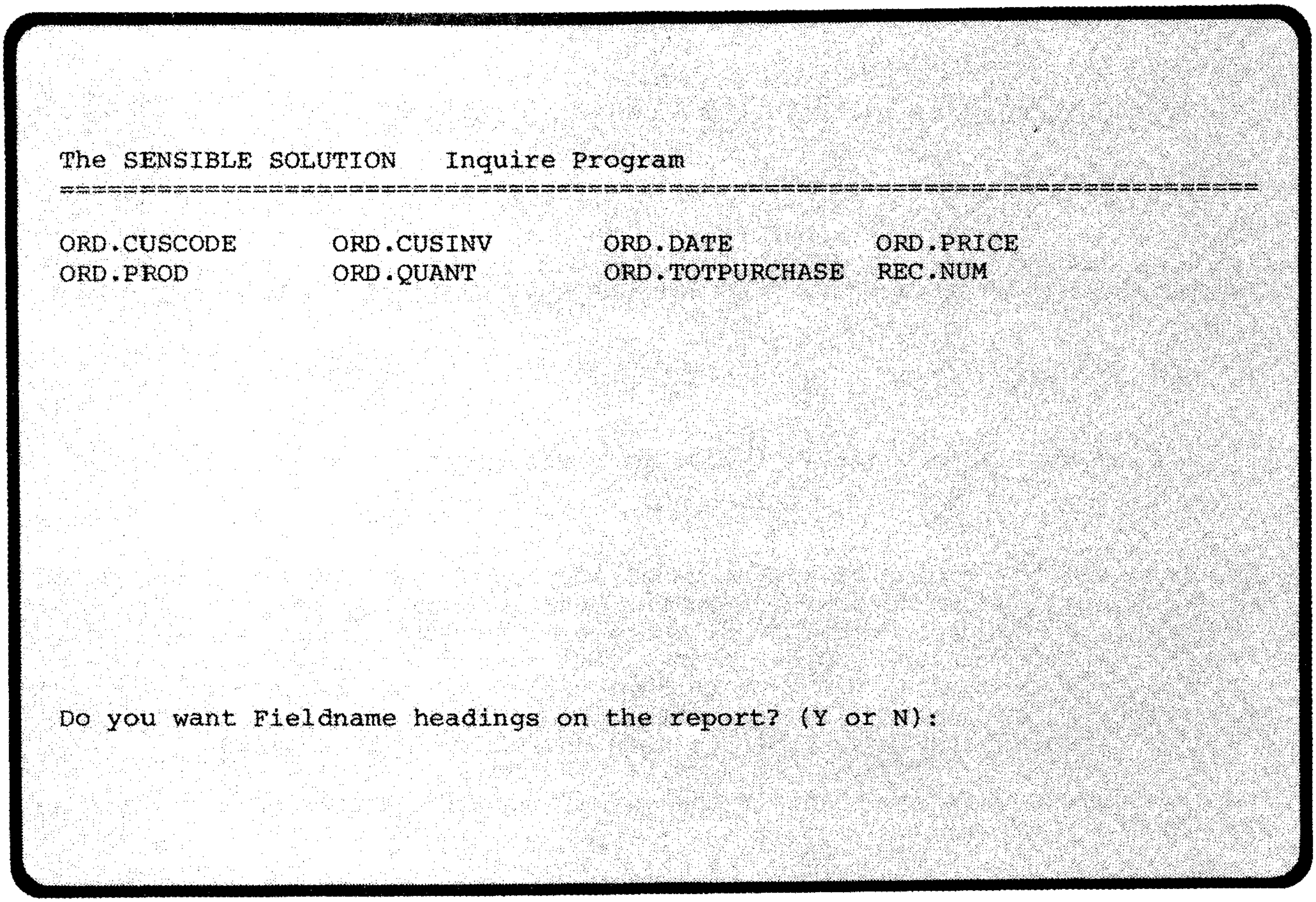
Now the cycle will begin and Inquire will request the name of a field you want on the report:

```
     Enter Fieldname to Print:
```

At this stage you should notice a directory displayed on the top of your screen that shows all of the fields contained in CUSTORD. You will also see a status line displayed on your screen:

```
     Total print-out length: ___ line ___
```

After selecting the file name **CUSTORD** --



```
The SENSIBLE SOLUTION   Inquire Program
===========================================================================

ORD.CUSCODE      ORD.CUSINV       ORD.DATE         ORD.PRICE
ORD.PROD         ORD.QUANT        ORD.TOTPURCHASE  REC.NUM










Do you want Fieldname headings on the report? (Y or N):
```

This line indicates the current position for the start of the next field you enter (columns) and the line number of your format. Remember, you can only specify a maximum of 4 lines as an Inquire display format. Do not, however, get this confused with the length of your report. A format defines how the print out will look for each record that is printed, and the report can be thousands of records in length.

Try pressing the **[ESC]** key and notice the high-lighted menu that appears at the bottom of your screen. Use the space bar to move the high-light block to any of the desired options. For now, though, turn the high-light menu off by pressing **[ESC]** again. We'll use some of the selections from this menu a little later on.

Now we'll begin specifying the field names that we want printed on our report and the headings that we want printed at the top of each column. At the prompt "Enter Fieldname to Print," type in our first field name -- ORD.CUSCODE. Now a new prompt will appear, "Enter substitute heading". Type in the words **Account Number** and after you press the **[RETURN]** key the "Enter Fieldname to Print" message will be displayed again. In this fashion we'll specify the next three fields that we want to see from CUSTORD and the column headings that we want printed at the top of the report. At the appropriate prompt enter the following information:

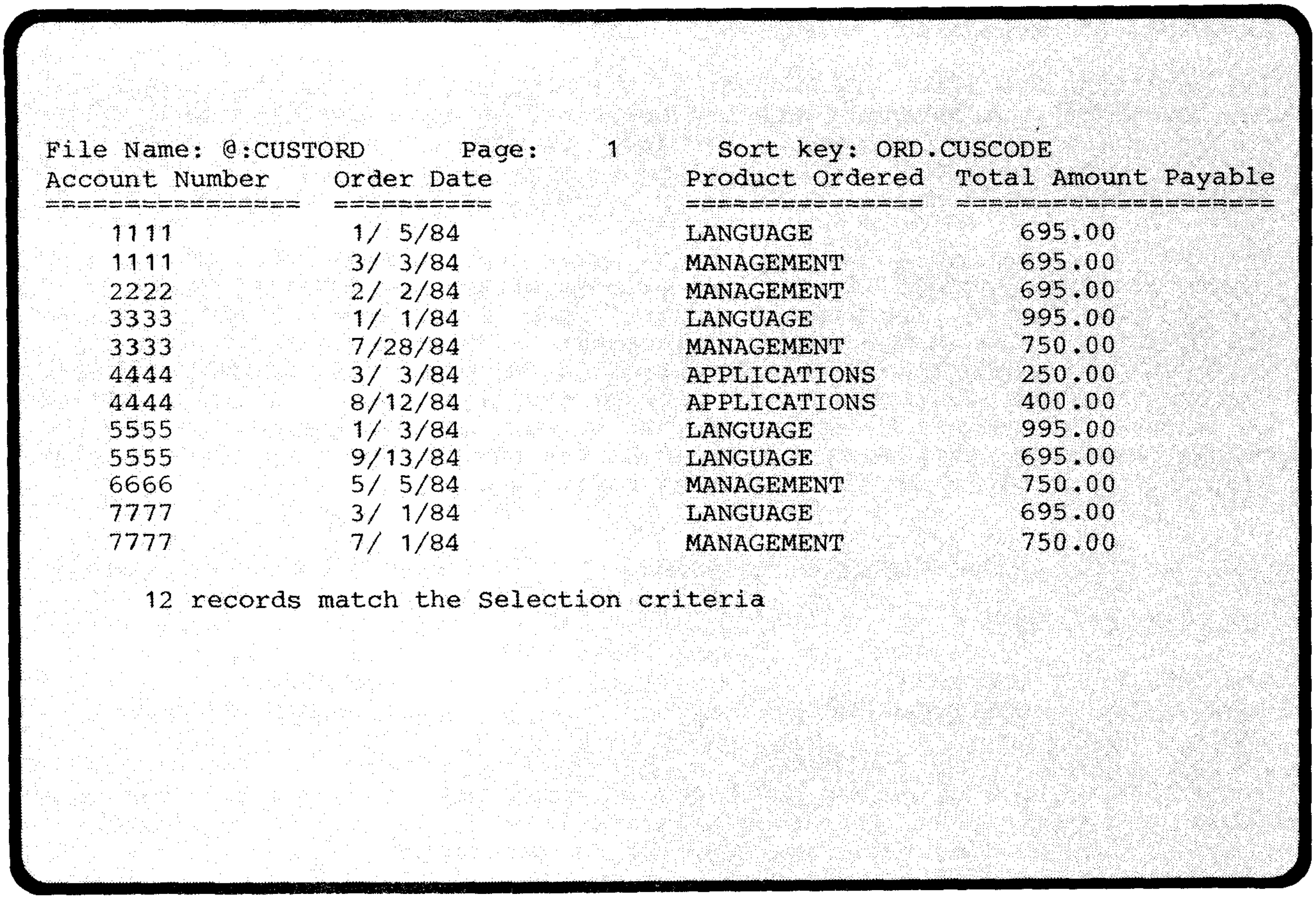
	Enter Fieldname to Print:	**Enter substitute heading:**
2nd. field:	ORD.DATE	Order Date
	Press **[ESC]** **[S]** (Space insert) and	specify 10 spaces.
3rd. field:	ORD.PROD	Product Ordered
4th. field:	ORD.TOTPURCHASE	Total Amount Payable

Everything we have discussed up to now is simply a repeating cycle that will allow you to define the format of a record as it is printed on our report -- field column, heading, line spacing, and column spacing.

Notice that every time that you specified a field, the directory displayed the field in half-intensity. When you are finished specifying the last column heading, "Total Amount Payable," you will again see the "Enter Fieldname to Print" prompt. This time just answer it with the **[RETURN]** key and we'll move on to specifying only those transactions that occur in 1984.

Enter Fieldname to Select by:

Inquire quick report of CUSTORD --



```
File Name: @:CUSTORD      Page:     1       Sort key: ORD.CUSCODE
Account Number       Order Date                   Product Ordered   Total Amount Payable
================     ==========                   ===============   ====================
    1111               1/ 5/84                    LANGUAGE                695.00
    1111               3/ 3/84                    MANAGEMENT              695.00
    2222               2/ 2/84                    MANAGEMENT              695.00
    3333               1/ 1/84                    LANGUAGE                995.00
    3333               7/28/84                    MANAGEMENT              750.00
    4444               3/ 3/84                    APPLICATIONS            250.00
    4444               8/12/84                    APPLICATIONS            400.00
    5555               1/ 3/84                    LANGUAGE                995.00
    5555               9/13/84                    LANGUAGE                695.00
    6666               5/ 5/84                    MANAGEMENT              750.00
    7777               3/ 1/84                    LANGUAGE                695.00
    7777               7/ 1/84                    MANAGEMENT              750.00

      12 records match the Selection criteria
```

The field name that you enter here will be used in a test that we are about to set up. We want to print out only those records that contain transactions made on or after January 1, 1984, so enter the date field, ORD.DATE.

```
=  is Equals   /  < is Less Than  / >  is Greater Than  / ! is Not Equal to
(  is Less Than or Equal to       / ) is Greater Than or Equal to
?  is If the Field Includes the Value/String Entered

What Operator do you wish to use:
```

Inquire will want to know how to select the records you desire. You just entered a field name, now use an operator to test a condition. Enter the right parentheses symbol, ")".

```
     What Value do you wish to use:
```

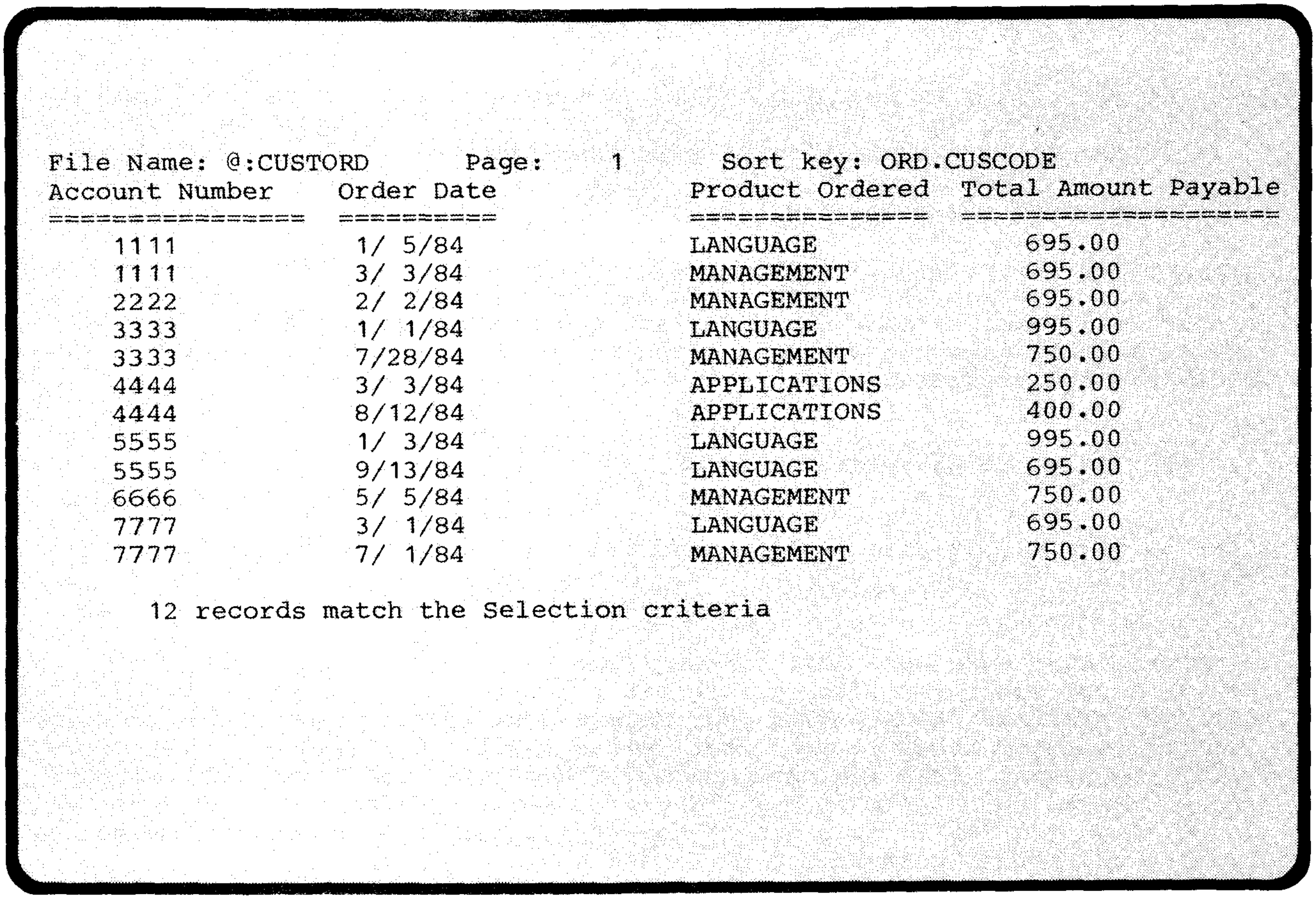
Since we specified a date field, you will see a date type of prompt. Enter "01/01/84".

We should also note here that you are free to select a field even if it is not part of the information printed in the report. You can also select multiple criteria thus making your report more restrictive. Your report will contain only those records that passed the first test, and the second test, and so on. When you have completed defining your selection criteria, simply press the **[RETURN]** key after the "Enter Fieldname to Select by" prompt and you will move out of this cycle and into the last phase of defining an Inquire format:

```
     Enter Fieldname to Sort by:
```

At the top of your screen you will see a directory of all key fields for your particular data file. SENSIBLE SOLUTION can only sort data by a key field. For our CUSTORD report let's sort the data into "Account Number" order. Enter the key field name ORD.CUSCODE.

Inquire quick report of CUSTORD --



```
File Name: @:CUSTORD      Page:     1       Sort key: ORD.CUSCODE
Account Number    Order Date            Product Ordered  Total Amount Payable
===============   ==========            ===============  ====================
    1111           1/ 5/84              LANGUAGE             695.00
    1111           3/ 3/84              MANAGEMENT           695.00
    2222           2/ 2/84              MANAGEMENT           695.00
    3333           1/ 1/84              LANGUAGE             995.00
    3333           7/28/84              MANAGEMENT           750.00
    4444           3/ 3/84              APPLICATIONS         250.00
    4444           8/12/84              APPLICATIONS         400.00
    5555           1/ 3/84              LANGUAGE             995.00
    5555           9/13/84              LANGUAGE             695.00
    6666           5/ 5/84              MANAGEMENT           750.00
    7777           3/ 1/84              LANGUAGE             695.00
    7777           7/ 1/84              MANAGEMENT           750.00

      12 records match the Selection criteria
```

```
     Send output to Printer, Crt, or Disk file? (Press P, C, or D)
```

There, we're finished specifying our quick report format for CUSTORD. All we have to do now is tell Inquire where we want to send the report. Since our report is less than 80 columns we will be able to see all of it on our display screen, so just press **[RETURN]** to accept the default, CRT. The Inquire report will immediately be displayed; it should look like the format on the facing page. Did you enter the 10 spaces between the second and third columns? As you recall, the menu that is displayed after pressing the **[ESC]** key at the "Enter Fieldname to Print" prompt will allow you to insert spaces and blank lines in your format. When we're finished, try creating some other reports with Inquire and experiment with inserting blank lines, displaying some fields on other lines, and inserting spaces between columns.

When the report is finished press the **[RETURN]** key and Inquire will show you our selection criteria and display format.

```
     File Selected: CUSTORD
     Heading Option: Substituted
     Field(s) Selected:
     Line 1 ORD.CUSCODE   ORD.DATE   ORD.PROD   ORD.TOTPURCHASE
     Selection criteria:
          When ORD.DATE is Greater Than or Equal to 01/01/84
     Sort Key Selected:  ORD.CUSCODE
```

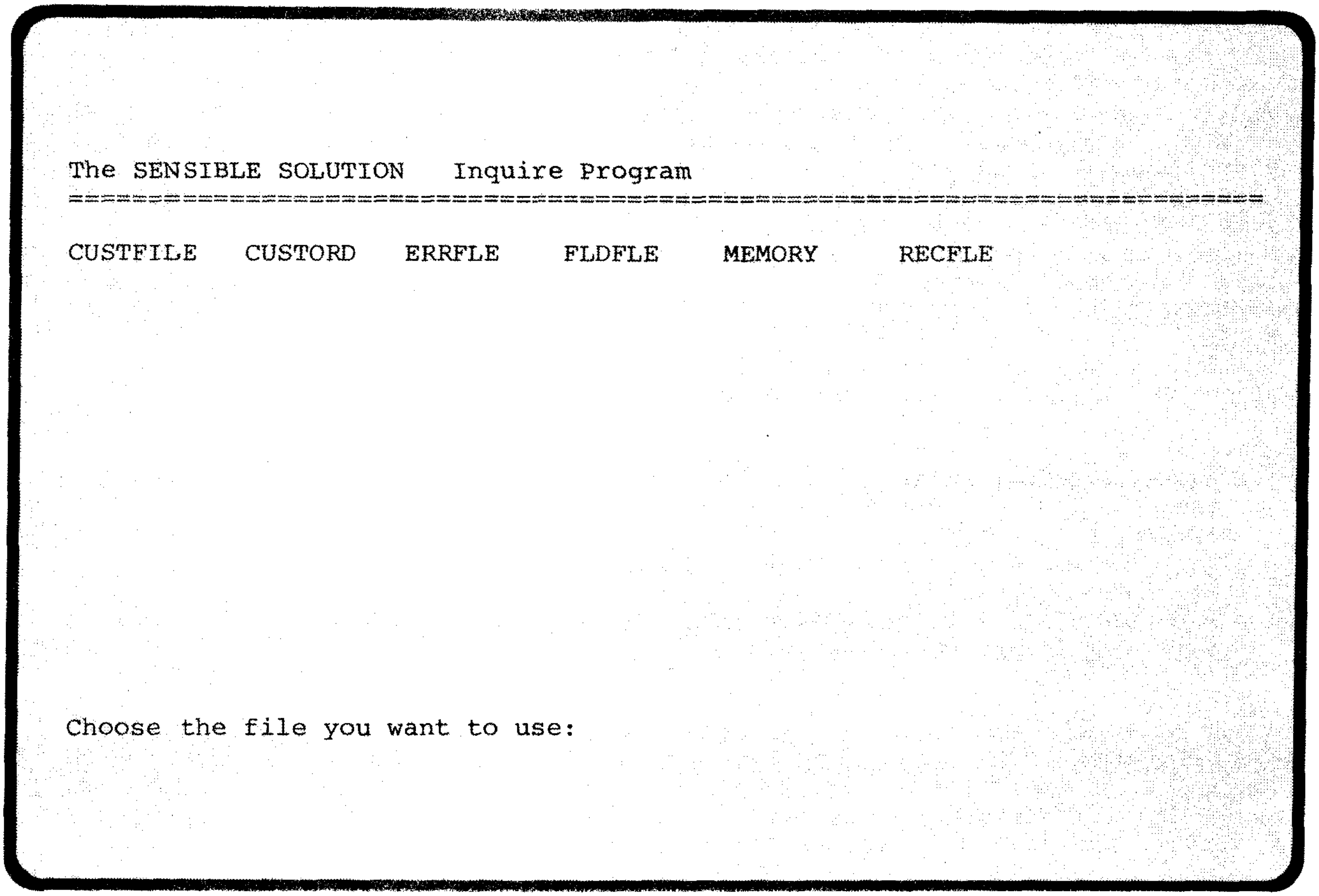
At the bottom of your screen a prompt will appear asking if you want to save this format. One of nicest features of Inquire is its ability to save formats in a disk file. To save it, answer "Y" to the question and then answer the next prompt with a file name. Your report format will be saved in a disk file with the extension ".IQ".

```
     Enter format Filename (save):
```

You will now be returned to the original data file directory where you can begin creating another Inquire quick report, exit back to the main menu or the operating system, or re-run any previously defined Inquire reports.

Now that you're back at the start of Inquire (you should see the complete file directory) press the **[ESC]** key and you'll get another high-lighted menu. Move the high-light block over the "Load Format" option and press

"Main Menu 10) Inquire" --

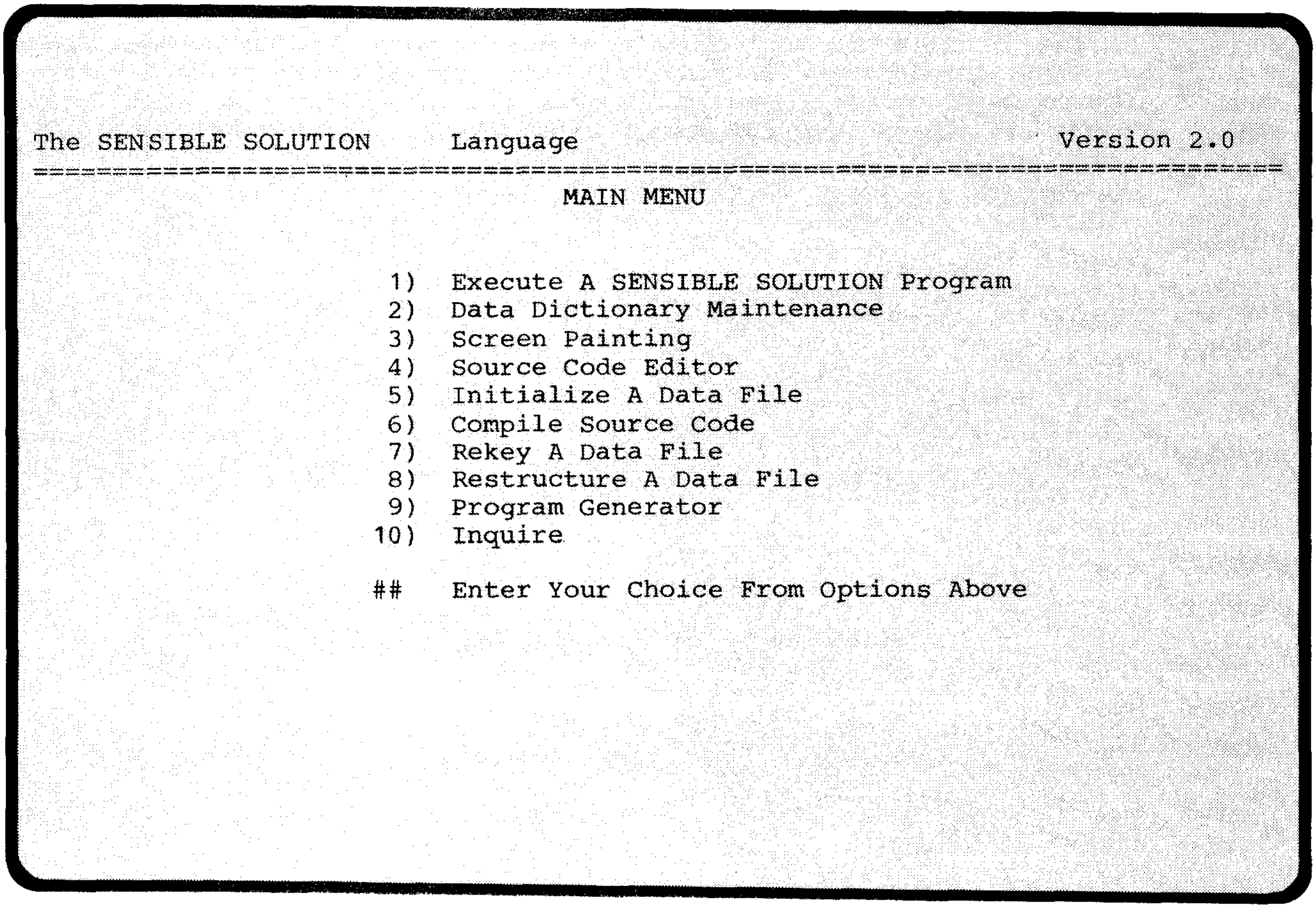


```
The SENSIBLE SOLUTION   Inquire Program
==========================================================================

CUSTFILE   CUSTORD   ERRFLE   FLDFLE   MEMORY    RECFLE










Choose the file you want to use:
```

the **[RETURN]** key. Now enter the file name where you saved our CUSTORD Inquire report format and we'll run the report again. First, the format criteria will be displayed on your screen and then you can go ahead and execute the report again.

As you can see, this a powerful feature of the Inquire program. You can experiment with developing really useful report formats and then, after settling on a good formula, save the format and run it any time that you want. You will always get the most current report from your SENSIBLE SOLUTION data base.

After typing SENSIBLE **[RETURN]** --



```
The SENSIBLE SOLUTION     Language                                Version 2.0
=============================================================================
                                   MAIN MENU


                          1)  Execute A SENSIBLE SOLUTION Program
                          2)  Data Dictionary Maintenance
                          3)  Screen Painting
                          4)  Source Code Editor
                          5)  Initialize A Data File
                          6)  Compile Source Code
                          7)  Rekey A Data File
                          8)  Restructure A Data File
                          9)  Program Generator
                         10)  Inquire

                         ##   Enter Your Choice From Options Above
```

LESSON 11
The Reporter Format

When called from the Main Menu, selection 3) Screen Painting offers the choice of two different formats. The REPORTER format does for printed reports what the SCREEN format does for a terminal display: it allows the interactive definition of a report format with headings and footings, detail and summary lines, calculated totals, and multiple-file sources of data.

Once defined, a reporter format may be used repeatedly in SENSIBLE SOLUTION programs in much the same way that screen formats are used. You create a reporter format by "screen painting". The reporter format can then be utilized by SENSIBLE SOLUTION programs to send formatted reports to your computer's printer. Creating reports with the reporter format involves the same operations used in creating any SENSIBLE SOLUTION program:

REPORTER APPLICATIONS

Paint Report Format
Edit Source Code File
Compile Commands
Execute Commands

Reporter formats differ slightly from screen formats.

A reporter format can be much larger than a screen format: a maximum of 255 columns wide and 60 lines in height. You're probably wondering how we can fit a 255-column print-page on an 80-column terminal display. Simple! We don't -- at least not all at once. The screen painting facility has a method for getting around the limitation of your screen size.

The Screen Painting menu option, **[ESC] [J] (Jump screen)** will allow you to shift back and forth between two views of the reporter screen. The left screen is defined as column 01 through column 79; the right screen is defined as column 49 through column 127. Thus, there is a 30 column overlap area from columns 49 through 79. This overlap area will make it easy for you to "keep your bearings" as you shift back and forth between the left-hand and right-hand portions of the report format screen.

The numbers on the left side of this page are to show you how we have counted the report format lines in screen painting this Reporter Format --

```
1                          OUR COMPANY SALES REPORT               PAGE ##
2-------------------------------------------------------------------------------
3 COMPANY NAME                                 ACCT. NUMBER       DATE STARTED
4-------------------------------------------------------------------------------
5                            SALES             OUTSTANDING        DATE OF LAST
6                             YTD              RECEIVABLES          PAYMENT
7-------------------------------------------------------------------------------
8***********************************           ########             mm/dd/yy
9                       #########.##           #########.##         mm/dd/yy
10        TOTAL SALES:  #########.##
11                          TOTAL RECEIVABLES: #########.##
```

```
 01                               79
  +--------------------------------+ (line 01)
  | (left side)                    |
  |                                |
  |                                |
  |                  49            |              127
  |                  +------------------------------+ (line 01)
  |                  |             |    (right side)|
  |                  |             |                |
  +--------------------------------+                |
                     |                              |
                     |                              |
                     |                              |
                     +------------------------------+
```

Use **[ESC] [J]** To Shift Views

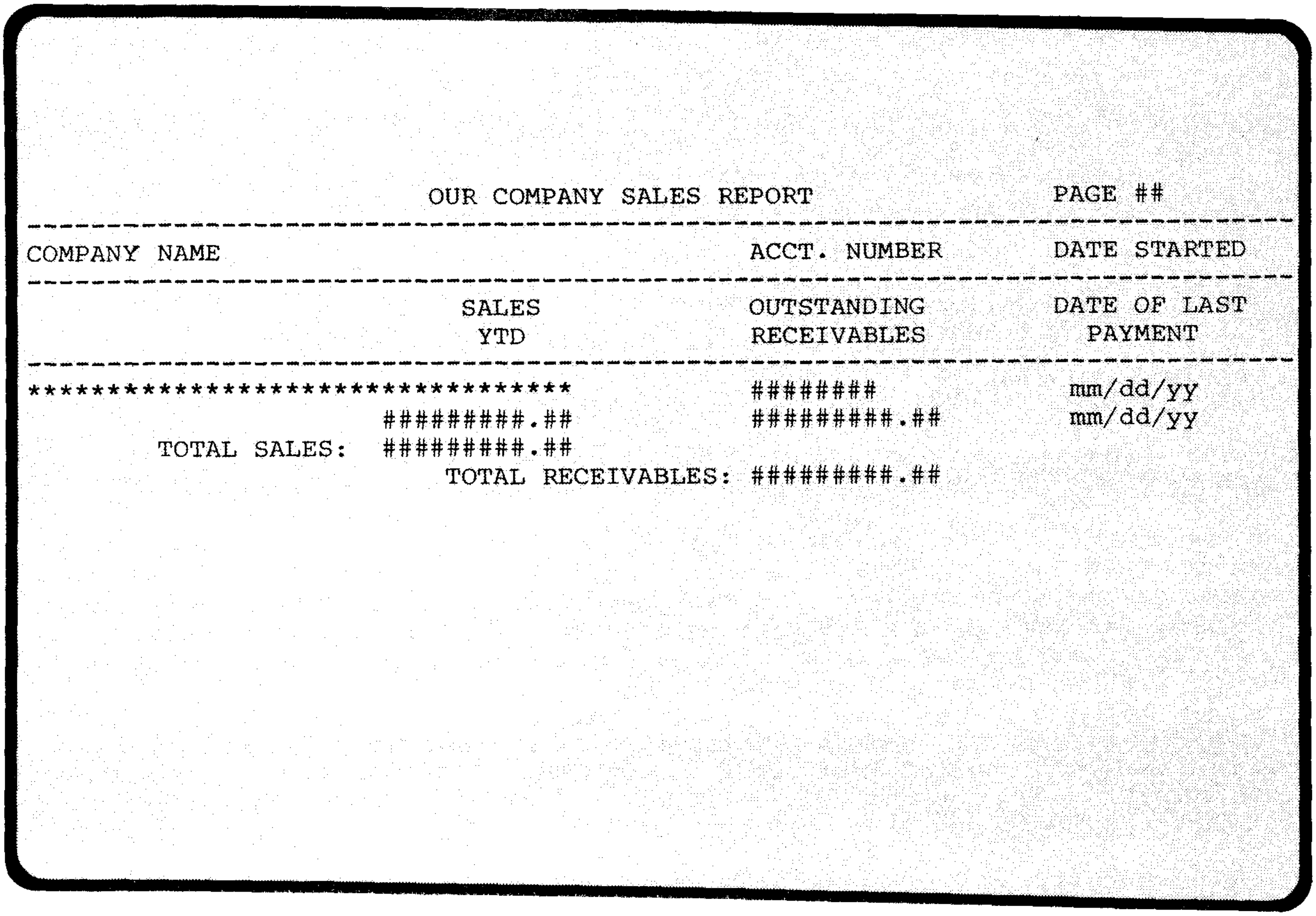
You'll find it becomes second nature to **[ESC] [J] (Jump screen)** back and forth across the 128-column page width as you're defining your format.

As we stated earlier, you can create report formats up to 255 columns wide. To extend a report format line beyond the 127th. column, simply place a comma in the 127th. column. The next line down on the format screen becomes columns 127 through 255 and will be attached to that line so that when the report is sent to the printer, the two different lines on your screen will be printed on the same line level on paper.

Each format line can contain printed information and field placements. When a format line is called from within a SENSIBLE SOLUTION program, the program collects the appropriate field values and prints them in the specified format for that line.

A reporter format may contain as many as 60 format lines. That does **not** mean that you can only print reports 60 lines long! Each format line in a reporter format may, and usually will, be used as the "template" for **several** lines of data on the actual report. For example, if you're printing a week-by-week listing of several data fields, you will probably use the same format line 52 times. The 60 format-line limit simply means that you can only have 60 different line formats appearing on a single report including: page headers, column headers, footings, column entries, totals, etc.. Frankly, we've never seen a report so complex that 60 format lines won't do the job. Most of your reports won't go over ten!

Screen paint this Reporter Format and **[ESC] [A]** add fields as displayed below --



```
                         OUR COMPANY SALES REPORT               PAGE ##
-------------------------------------------------------------------------------
COMPANY NAME                                 ACCT. NUMBER       DATE STARTED
-------------------------------------------------------------------------------
                           SALES             OUTSTANDING        DATE OF LAST
                            YTD              RECEIVABLES          PAYMENT
-------------------------------------------------------------------------------
**********************************           ########            mm/dd/yy
                      #########.##           #########.##        mm/dd/yy
        TOTAL SALES:  #########.##
                          TOTAL RECEIVABLES: #########.##
```

The best way to learn to use reporter formats is by example so we're going to create a sample Sales Report for our old friend, **CUSTFILE.** We will have to perform five complete steps to produce an executable program to generate sales reports. Each step will involve a separate program from the SENSIBLE SOLUTION main menu.

STEP ONE -- **Use 3) Screen Painting**

Specify that you are creating a **Reporter format.** Enter the name of the format: **SALES.** Create the report format lines as you see them on the opposite page. Your control keys operate just as they do when creating screen formats, with this exception: you can not have boxes in a reporter format. Move the cursor, and type headings and labels. Now use **[ESC] [A] (Add field)** to place each of the fields to be shown on the report.

Here is a partial list of the field name entries you will need to "Add" to your report. Do so now.

COMPANY NAME = **CUS.NAME** (place field on line #8)
ACCOUNT NUMBER = **CUS.CUSCODE** (place field on line #8)
DATE STARTED = **CUS.DATE** (place field on line #8)
SALES YTD = **CUS.SALES** (place field on line #9)
OUTSTANDING RECEIVABLES = **CUS.RECEIVE** (place field on line #9)
DATE LAST PAID = **CUS.LASTPYMT** (place field on line #9)

In addition to the fields from **CUSTFILE**, there are three fields we will need to provide values for the format lines -- "PAGE" (line #1), "TOTAL SALES" (line #10), and "TOTAL RECEIVABLES" (line #11).

These fields must be defined in the Data Dictionary **before** we can make reference to them on a report format. **[ESC] [A] (Add field)** will allow us to add the definitions to the dictionary.

All three of these fields are "temporary", in that the information they hold will not be retained in a data file after the report has been printed. So, we will place them in the "phantom file", **MEMORY**, we created for the **CUSTPYMT** screen. We'll expand MEMORY to accommodate the new temporary storage fields we'll need.

The names of these fields follow the naming convention we discussed earlier:

N.12.2.1 and N.12.2.2

"N" means that the field holds a numeric value.
"12" refers to the actual number of characters that may be entered to the field (it's length).

Use **ESC] [H] (Hard copy)** to obtain the following print-out --

```
@:SALES   .SCC reporter format listing      Page No: 0001

                         OUR COMPANY SALES REPORT               PAGE ##
--------------------------------------------------------------------------------
COMPANY NAME                                 ACCT. NUMBER       DATE STARTED
--------------------------------------------------------------------------------
                           SALES             OUTSTANDING        DATE OF LAST
                            YTD              RECEIVABLES          PAYMENT
--------------------------------------------------------------------------------
**********************************           ########            mm/dd/yy
                      #########.##           #########.##        mm/dd/yy
        TOTAL SALES:  #########.##
                          TOTAL RECEIVABLES: #########.##


SALES   .SCC reporter format listing      Page No: 0002

Field name       File     Size   Col  Row  Key
---------------  -------- ----   ---  ---  ---
N.2.0.1          MEMORY      2   070   01   N
CUS.NAME         CUSTFILE   34   001   08   Y
CUS.CUSCODE      CUSTFILE    8   046   08   Y
CUS.DATE         CUSTFILE    8   066   08   N
CUS.SALES        CUSTFILE   12   023   09   N
CUS.RECEIVE      CUSTFILE   12   046   09   N
CUS.LASTPYMT     CUSTFILE    8   066   09   N
N.12.2.1         MEMORY     12   023   10   N
N.12.2.2         MEMORY     12   046   11   N
```

"2" means that there can be two decimal places in the number.
"1" and "2" (or 3 or 4 or 5, etc) allow us to differentiate between memory fields that are the same size.

Position your cursor appropriately and add these fields:

TOTAL SALES =	Field Name:	N.12.2.1 (place field on line #10)
	File Name:	MEMORY
	Type:	N
	Length:	12
	Decimal:	2
	Key:	N
TOTAL RECEIVABLES =	Field Name:	N.12.2.2 (place field on line #11)
	File Name:	MEMORY
	Type:	N
	Length:	12
	Decimal:	2
	Key:	N
PAGE =	Field Name:	N.2.0.1 (place field on line #1)
	File Name:	MEMORY
	(this field was created previously)	

When the report is laid out correctly, send a copy of the format to the printer using **[ESC] [H] (Hard copy)**, then press **[ESC] {Q]** and save the format. The language Main Menu will return.

STEP TWO -- Use **(5.) Initialize A Data File**

You added two new fields to the file MEMORY. The SENSIBLE SOLUTION must be alerted to the change in file structure. Because MEMORY is a temporary storage file, it never retains "live" data. Consequently, we use "Main Menu 5) Initialize a Data File" to perform this function.

STEP THREE -- Use **(4.) Source Code Editor**

We must now create a program that will use this report format. Enter the Source Code Editor and for file name specify **SALES.** Now create the program source code file as shown on the next page. You'll want to keep your printout of the format at hand for reference because you'll need to know which format lines to use when you write SALES.

STEP FOUR -- Use **(6.) Compile a Source Code File**

Now simply "escape" back to the Main Menu and use selection 6 to compile

Use the Source Code Editor to create the **SALES** program listed below --

SALES   .SRR source file listing

```
            remark
            remark OPEN FILES:  CUSTFILE AND MEMORY
            trap PAGE BREAK gosub HEADING
            trap FILE ERROR goto TOTALS
            mount report format SALES print on ask at run time
            print page printable lines = 60
            print page total lines = 66
            find first rec using field CUS.CUSCODE
            gosub HEADING
START       print format-line# 08
            print format-line# 09
            N.12.2.1 = (N.12.2.1)+(CUS.SALES)
            N.12.2.2 = (N.12.2.2)+(CUS.RECEIVE)
            print blank lines 01
            find next rec in file CUSTFILE
            goto START
HEADING     N.2.0.1 = (N.2.0.1)+<1>
            print blank lines 01
            print format-line# 01
            print format-line# 02
            print format-line# 03
            print format-line# 04
            print format-line# 05
            print format-line# 06
            print format-line# 07
            print blank lines 01
            return
TOTALS      print format-line# 10
            print format-line# 11
            print page eject
            execute .Run file CUSMENU
```

your source code.

STEP FIVE -- Use **(1.) Execute A SENSIBLE SOLUTION Program**

When the main menu appears again you have a sales report program ready to generate your report. Choose selection 1 and specify the file name, SALES. THERE! You've done it.

Now that you've finished, let's go back and review the construction of our source code listing.

As we have done in our other programs, the first thing to do is to set **TRAPS** for certain conditions that may occur during program execution.

Line 3 **trap PAGE BREAK gosub HEADING**

Each time the printer reaches the end of a page, we want the program to "trap PAGE BREAK" and "gosub HEADING." The "gosub" jumps to the group of command lines labeled "HEADING." This subroutine will print our report heading at the top of the next page.

Line 4 **trap FILE ERROR goto TOTALS**

When the sales report has gathered all required information on each of the customers in CUSTFILE, an "end of file error" will occur. At this time, we will want to have our final sales and receivables totals printed and exit from the SALES program. So control branches to the portion of the program that does this (beginning with the line labeled TOTALS).

Line 5 **mount report format SALES print on ask at run time**

As in other programs we have created, we must "Mount" our SALES report format screen. At the time of program execution, we want the program to ask the operator where they want the report to be sent-- to the Disk, the CRT, or the Printer.

Line 6 **print page printable lines = 60**
Line 7 **print page total lines = 66**

The next two lines of our SALES program set the paper length at 66 lines and the amount of printable space on that paper at 60 lines.

Line 8 **find first rec using field CUS.CUSCODE**
Line 9 **gosub HEADING**

Next, we have the computer find the first record in CUSTFILE and then

Use the Source Code Editor to create the **SALES** program listed below --

SALES .SRR source file listing

```
           remark
           remark OPEN FILES:  CUSTFILE AND MEMORY
           trap PAGE BREAK gosub HEADING
           trap FILE ERROR goto TOTALS
           mount report format SALES print on ask at run time
           print page printable lines = 60
           print page total lines = 66
           find first rec using field CUS.CUSCODE
           gosub HEADING
START      print format-line# 08
           print format-line# 09
           N.12.2.1 = (N.12.2.1)+(CUS.SALES)
           N.12.2.2 = (N.12.2.2)+(CUS.RECEIVE)
           print blank lines 01
           find next rec in file CUSTFILE
           goto START
HEADING    N.2.0.1 = (N.2.0.1)+<1>
           print blank lines 01
           print format-line# 01
           print format-line# 02
           print format-line# 03
           print format-line# 04
           print format-line# 05
           print format-line# 06
           print format-line# 07
           print blank lines 01
           return
TOTALS     print format-line# 10
           print format-line# 11
           print page eject
           execute .Run file CUSMENU
```

program control jumps to the line labeled HEADING.

```
     Line 17 HEADING    N.2.0.1 = (N.2.0.1)+<1>
     Line 18            print blank lines  01
     Line 19            print format-line# 01
     Line 20            print format-line# 02
     Line 21            print format-line# 03
     Line 22            print format-line# 04
     Line 23            print format-line# 05
     Line 24            print format-line# 06
     Line 25            print format-line# 07
     Line 26            print blank lines  01
     Line 27            return
```

The "HEADING" subroutine prints the heading at the top of each page. N.2.0.1 = (N.2.0.1)+<1> is the command which specifies page number incrementation. Then we print a blank line, and next, the first seven lines of the reporter format which we created with the Screen Painter. This is followed by another blank line. Finally, program control is returned to the line following the "gosub HEADING" command.

```
     Line 10 START      print format-line# 08
     Line 11            print format-line# 09
     Line 12            N.12.2.1 = (N.12.2.1)+(CUS.SALES)
     Line 13            N.12.2.2 = (N.12.2.2)+(CUS.RECEIVE)
     Line 14            print blank lines  01
     Line 15            find next rec in file CUSTFILE
     Line 16            goto START
```

Beginning at the label START, the first two command lines (lines 10 and 11) print the two lines in our reporter format that contain the fields that hold individual customer information (i.e., company name, acct. number, date started, sales YTD, outstanding receivables, and date of last payment). Command lines 12 and 13 work as summing devices to compute TOTAL SALES and TOTAL RECEIVABLES. Line 14 prints a blank line, which will separate the data on each individual customer, making it easier to read our report. Now our program directs the computer to find the next customer record in CUSTFILE and "goto" the line labeled START. This program loop will continue until an "end of file" error occurs and is trapped by command line 4. The trap will send program control to the command line labeled TOTALS.

```
     Line 28 TOTALS     print format-line# 10
     Line 29            print format-line# 11
     Line 30            print page eject
     Line 31            execute .Run file CUSMENU
```

Use the Source Code Editor to create the **SALES** program listed below --

SALES   .SRR source file listing

```
           remark
           remark OPEN FILES:  CUSTFILE AND MEMORY
           trap PAGE BREAK gosub HEADING
           trap FILE ERROR goto TOTALS
           mount report format SALES print on ask at run time
           print page printable lines = 60
           print page total lines = 66
           find first rec using field CUS.CUSCODE
           gosub HEADING
START      print format-line# 08
           print format-line# 09
           N.12.2.1 = (N.12.2.1)+(CUS.SALES)
           N.12.2.2 = (N.12.2.2)+(CUS.RECEIVE)
           print blank lines 01
           find next rec in file CUSTFILE
           goto START
HEADING    N.2.0.1 = (N.2.0.1)+<1>
           print blank lines 01
           print format-line# 01
           print format-line# 02
           print format-line# 03
           print format-line# 04
           print format-line# 05
           print format-line# 06
           print format-line# 07
           print blank lines 01
           return
TOTALS     print format-line# 10
           print format-line# 11
           print page eject
           execute .Run file CUSMENU
```

The group of command lines labeled TOTALS prints the sums of what was computed and stored by command lines 12 and 13. Format line 10 contains the line label "**TOTAL SALES**" and the memory field **N.12.2.1** which holds this value. Format line 11 contains the line label "**TOTAL RECEIVABLES**" and the memory field **N.12.2.2** which holds that value. Since the report is completed, the page eject command will immediately force a printer page advance, based on the number of lines we specified are to be printed on each page. If the program user sent the report to the screen instead of to the printer, this command will have the effect of locking the screen display until the user presses the **[RETURN]** key. Without this command, the program would exit immediately and the user would not have the opportunity to view the information in the report. Finally, the program ends by executing the .RUN file CUSMENU, and program users are presented the CUSMENU to allow for further menu selections.

Graduation Day!

The Reference Section of the SENSIBLE SOLUTION manual contains detailed information on all the features, procedures and commands in the language. In addition you will find discussions on file and record locking on multi-user systems and converting foreign databases into data structures that can be read and manipulated by SENSIBLE SOLUTION.

What we've given you is an introduction to a system that lets you quickly, interactively design and modify business and database applications. The SENSIBLE SOLUTION keeps track of the frustrating details while you concentrate on designing new applications, enhancing old ones, and integrating applications into your particular business environment.

Is it any wonder we call it The SENSIBLE SOLUTION?

* * *

The SENSIBLE SOLUTION Language

Table of Contents

REFERENCE MANUAL

MENU SELECTIONS

LANGUAGE COMMANDS

DATA STRUCTURES

APPENDIX

The SENSIBLE SOLUTION Language

FILE EXTENSIONS

filename.COM	Directly executable machine code file
filename.SCC	Screen format or report format source file
filename.SRR	Source code file
filename.RUN	Executable SENSIBLE SOLUTION program
filename.MS	Master data file
filename.KS	Key file (for the .MS file)
filename.IQ	Inquire report format and selection criteria file
filename.LST	ASCII text file

The SENSIBLE SOLUTION Language

FILE LIST

MAILLIST.RUN	mail list demonstration program
MAILLIST.KS	key file
MAILLIST.MS	data file
MAILLIST.SCC	screen format file
MAILLIST.SRR	source code file
MENU .RUN	main menu program
MENU .SCC	screen format file
MENU .SRR	source code file
ERRENT .RUN	change system error message program
ERRFLE .KS	error message key file
ERRFLE .MS	error message data file
FLDFLE .KS	key file of all field information
FLDFLE .MS	data file of all field information
RECFLE .KS	key file of all file information
RECFLE .MS	data file of all file information
MEMORY .KS	temporary memory variable key file
MEMORY .MS	temporary memory variable data file
SENSETUP.COM	system installation/configuration program
SENSCTRL.MS	system definition data file
TERMDEFS.MS	terminal definition data file
SENSFREE.COM	multi-user utility program
SENSIBLE.COM	language executive program
ENTFLE .RUN	data dictionary maintenance program
SENSCRN .COM	screen painting program
SENSCMD .COM	source code editor program
SENSINIT.COM	file initialization program
SENSCOMP.COM	compiler program
SENSRKEY.COM	key file re-key program
SENSRSTC.COM	data file restructure program
SENSGEN .COM	automatic program generator program
SENSINQR.COM	quick report generator program

The SENSIBLE SOLUTION Language

MAIN MENU PROGRAMS

1)	SENSIBLE.COM	Execute A SENSIBLE SOLUTION Program
2)	ENTFLE .RUN	Data Dictionary Maintenance
3)	SENSCRN .COM	Screen Painting
4)	SENSCMD .COM	Source Code Editor
5)	SENSINIT.COM	Initialize A Data File
6)	SENSCOMP.COM	Compile A Source Code File
7)	SENSRKEY.COM	Rekey A Data File
8)	SENSRSTC.COM	Restructure A Data File
9)	SENSGEN .COM	Program Generator
10)	SENSINQR.COM	Inquire

you want a list of all zip codes in this file that are greater than or equal to "90000". After you enter the operator symbol, you will immediately be prompted with:

What Value do you wish to use:

Enter the number "90000". When the report is finally run, all records in the data file that contain a ZIP.CODE value of 90000 or greater will be listed out.

Notice that you are free to select a field even if it is not part of the information printed in the report. You can also select multiple criteria thus making your report more restrictive. Your report will contain only those records that passed the first test, and the second test, and so on. When you have completed defining your selection criteria, simply press the [RETURN] key after the "Enter Fieldname to Select by" prompt and you will move out of this cycle and into the next phase of defining an Inquire format:

Enter Fieldname to Sort by:

At the top of your screen you will see a directory of all key fields for your particular data file. Enter one of these field names to give Inquire a sorting criteria for your report.

Inquire will now ask where you want the report printed out:

Send output to Printer, Crt, or Disk file? (Press P, C, or D)

If you elect to send it to a disk file your report can later be read by a text editor or word processing program for inclusion in a document or letter. When the report print out is finished you will see a summary of your selection criteria and display format. If you chose to have the report sent to your screen, press the [S] key or [RETURN] to indicate to the program that your work is complete and you will see the criteria and format information:

File Selected:
Heading Option:
Field(s) Selected:

Line 1
Line 2
(up to 4 lines)
Selection criteria:
Sort Key Selected:

At the bottom of your screen a prompt will appear asking if you want to save this format. One of nicest features of Inquire is its ability to save frequently used formats in a disk file. To save it, answer "Y" to the question and then answer the next prompt with a file name. Your report format will be saved in a disk file with the extension ".IQ".

Enter format Filename (save):

You will now be returned to the original data file directory where you can begin the creation of another Inquire quick report, exit back to the main menu or the operating system, or re-run any previously defined Inquire reports. Remember, Inquire reports are all named "filename.IQ". Use your operating system directory to keep track of all the Inquire report formats you develop.

Any time that the first data file directory is displayed by Inquire, you have the option of either choosing one of the displayed file names to create a new quick report or you can press your [ESC] key to load and run a previously defined report format. If you choose the latter, use "Load format" from the high-lighted menu and then enter the file name (filename.IQ) of the format you wish to use. The format that you specified will be displayed on your screen. You can then examine the format criteria and, if you are satisfied, run the report.

file name from which you wish to create a quick report. Next, you will see the following prompt:

Do you want Fieldname headings on report? (Y or N):

In a moment you will be selecting the field data that will be printed on the report in columnar form. Answering "Y" to this prompt will force Inquire to print a column heading that will appear at the top of every page of the report as it is printed out.

The next prompt,

Do you wish to substitute Fieldname headings? (Y or N):

will give you the option of either using the actual field names as headings on your report or substituting a unique heading that you will specify. If you answer "Y" to the above prompt, every time that you select a field to be printed on the report, another prompt will soon follow requesting that you specify whatever heading you want to see printed at the top of each page of the report. Inquire will ask for a field then Inquire will ask for a heading.

Make your choice and you will then be requested to enter the name of a field you want on the report:

Enter Fieldname to print:

At this stage you should notice a directory displayed on the top of your screen that shows all of the fields contained in this data file. You will also see a status line displayed on your screen:

Total print-out length: ___ line ___

This line indicates the current position for the start of the next field you enter (columns) and the line number of your format. Remember, you can only specify a maximum of 4 lines as an Inquire display format. Do not, however, get this confused with the length of your report. A format defines how the print out will look for each record that is printed, and the report can be thousands of records in length.

If you press the [ESC] key at this stage, a high-lighted menu will appear at the bottom of your screen. Use the space bar to move the high-light block to the desired option and press the [RETURN] key to activate the option. If your field name directory is very long, you can use "Next screen" and "Previous screen" to scroll through it. "Line change" will shift you to another format line (from 1 to 4), "Space insert" will allow you to create spaces in between your columns (2 are automatically inserted), "Restart program" will return you to the original data file directory, and "Quit" will abort the Inquire program and return you to the operating system or the main menu. To turn off this six option menu and return to the prompt, "Enter Fieldname to print," simply press the [ESC] key again.

Everything we have discussed up to now is simply a repeating cycle that will allow you to define the format of a record as it is printed on your report -- field column, heading, line spacing, and column spacing.

Every time that you specify a new field, the directory will display the field in half-intensity. When you are finished specifying all of the fields that you want printed on the report, answer the "Enter Fieldname to print" prompt with a carriage return -- [RETURN].

Inquire will now begin another phase of the report format definition:

Enter Fieldname to Select by:

The field name that you enter here will be used in a test that you are about to set up:

= is Equals / < is Less Than / > is Greater Than / ! is Not Equal to

(is Less Than or Equal to /) is Greater Than or Equal to

? is If the Field Includes the Value/String Entered

What Operator do you wish to use:

Inquire will want to know how to select the records you desire. You just entered a field name, now use an operator to specify some value for all of the records. For example, if you selected a zip code field called ZIP.CODE, you could then choose the operator ")" to indicate to Inquire that

This running comment alerts the operator that the program generator is working. Note that the generator may pause for several seconds when integrating a screen format file into the compiled command file. This is because each field reference in a screen or report must be located and linked.

If any errors are encountered during program generation, an appropriate message is displayed. All error messages are self-explanatory. A complete list of error messages and explanatory comments will be found in the Appendix of this manual. When SENSGEN is finished, it will return you to the system level or the main menu depending upon how you initiated it.

If, when you enter the file name, SENSGEN finds that a matching program source code file (filename.SRR) already exists on the disk, it will ask if you wish to overwrite that source code file -- "Overwrite?". If you answer "Y", the automatically compiled source code file (filename.SRR) will replace the older source file bearing the name "filename.SRR." The older files will be lost.

Inquire

Overview:

"Inquire" (SENSINQR.COM) is a quick, powerful, data-scanning facility provided with SENSIBLE SOLUTION. Inquire will allow you to extract data from a single file; create a unique display format; send the formatted report to the CRT, disk drive, or printer; and then save the display format in a disk file for future use. You can use any combination of fields to define a selection criteria for a report. Each record that meets the selection criteria that you specify can be printed out on from 1 to 4 lines on the report. The fields that you specify will form the columns of the report. You can adjust the spacing between the columns and you can also print blanks lines between the records. Inquire reports can extend to a maximum width of 127 columns. The Inquire quick report generator also provides an excellent tool for experimenting with formats that you may later wish to incorporate into a SENSIBLE SOLUTION report generating program.

Operation:

To execute Inquire from the operating system level type you can type out a command line:

```
d>SENSINQR d:formatname [RETURN]
```

where "d" is the drive location of the format, and formatname is the name of a format which you have previously created and saved using SENSINQR. Following execution, the program will return to the operating system level.

If you choose to initiate Inquire by typing SENSINQR from the system level or by choosing selection 10 from the main menu, Inquire will begin by displaying a directory of all data files and then request that you enter a

If the data file that you specified in the above message can not be found, another message will be displayed:

________ not found in Data Dictionary

After entering a valid file name and drive location, RESTRUCTURE will immediately begin execution. It will display five messages, one after the other, describing each phase of operation as it reads the old data file and translates it into the newly created file. When the program is finished, a message will appear to remind you to REKEY the file and RECOMPILE all programs that access the data file.

The RESTRUCTURE program involves a great deal of disk accessing, particularly with a large data file having many keys. For this reason, RESTRUCTURE is time consuming. If you have a multi-user system with large data files, we recommend that a restructure operation be done when your computer system is quiet, such as overnight.

One other multi-user consideration: you can not restructure a data file that is currently being accessed by another user. However, the corollary is also true -- once you have successfully initiated RESTRUCTURE, any other user attempting to access the same data file will be "locked out".

Program Generator

Overview:

This program will read a specified screen format (filename.SCC) and then automatically create a file maintenance program source code listing (filename.SRR) and an executable SENSIBLE SOLUTION program (filename.RUN). If the data files (filename.MS/.KS) referenced by the screen format have not yet been initialized, it will also automatically initialize the data files prior to compiling the program. SENSGEN.COM will accept a maximum of sixteen data files. The resulting file maintenance program may be used immediately to examine, update, or enter new records in a data file. It may also be modified and extended with the source code editor, SENSCMD.COM, to quickly develop a more sophisticated application program.

Operation:

SENSGEN.COM may be initiated from either the operating system level or from the main menu. The program will begin by asking you the name of the screen format file (filename.SCC) you want it to read. Enter that name and SENSGEN will create source code for a file maintenance program (filename.SRR), initialize the appropriate data files if they do not exist (filename.MS/.KS), and compile the source code into an executable program (filename.RUN). The SENSIBLE SOLUTION program generator goes through five phases:

-- Checking fields
-- Initializing Data files
-- Generating Command Source file
-- Checking for Target Labels
-- Checking for Goto/Gosubs

... and for each phase tells which line of the command file it is scanning.

Restructure A Data File

Overview:

It often happens, as an application is put to use, that the structure of the data file must change. New fields are needed, old fields are seen to be unnecessary, some fields must stretch to accept more characters or shrink to conserve disk space. SENSRSTC.COM, the restructure utility, provides a convenient way of converting existing data files to the new structure without losing any of the data.

SENSRSTC.COM performs three operations:

1.) SENSRSTC reads FLDFLE (the Data Dictionary) to determine what changes you have made to the current file structure definition. That is, it reads the field names, location of fields within the record ("Offset"), the type of fields (alphabetic, numeric, date, etc.) and whether you have added or deleted keys. Then it compares this old definition to the new definition.

2.) SENSRSTC calculates the new "Offset" values -- the location of each field within the record -- and embeds the new "Offset" values in FLDFLE.

3.) It reads the original data file (filename.MS --the file containing the original data that you wish to preserve), one record at a time, and rewrites the data out to a new filename.MS using the new file structure definition.

You need only restructure data files that have records in them that you wish to retain. If you are willing to abandon the data in a file, you need merely INITIALIZE the file (using SENSINIT.COM) after you have made the necessary changes in the Data Dictionary. Remember, though, **you must never alter the structure and then INITIALIZE a data file that contains data that you wish to preserve.** A complete discussion on when to use the INITIALIZE, REKEY, and RESTRUCTURE programs can be found in the reference section on Data

Structures.

Operation:

SENSRSTC.COM, the RESTRUCTURE program can be run from either the operating system level or from the SENSIBLE SOLUTION main menu. If it is executed from the operating system level, you can type out a command line:

```
d>SENSRSTC filename ###
```

where filename is the name of the data file (you do not need to specify the file extension) and, optionally, "###" is the number of records you want to restructure. Following program execution, you will be returned to the system level.

If you choose to initiate RESTRUCTURE by typing SENSRSTC from the system level or by selecting main menu number 8, "Restructure A Data File," the following prompt will be displayed on your screen:

```
Enter the name of the file to be restructured  ********
```

Enter the name of the data file (filename.MS), and the following message will be displayed:

```
Create work file filename.M$$ on drive _: (press [ENTER] or work drive letter)

(Restructure will abort if work drive has to little free space)
```

RESTRUCTURE has to make a temporary work file before it creates the new restructured data file. Thus, for a time, your disk storage will have to contain an amount of data that is twice the size of the original data file. To provide for this extra work space, RESTRUCTURE will allow you to designate a different drive location to contain the temporary work file. For example, if your original data file is 1000K bytes in size, you will need at least 2000K bytes of free space on this same drive to avoid the necessity of designating a unique temporary work drive location. If your present drive location does not have enough free space, type in a different drive location letter than the one displayed.

Rekey A Data File

Overview:

Any time that a different set of key fields has been defined in an existing data file, the index-key file, filename.KS, must be rebuilt and the record counter in the file must be reset to reflect the total number of active records in the file. SENSRKEY.COM performs these functions by re-keying the SENSIBLE SOLUTION data file, filename.KS. SENSRKEY.COM should be be used any time that an existing data file containing "live" data has been restructured (see SENSRSTC.COM) and the number or length of key fields specified in the file has been changed.

If you have initialized an existing data file (see SENSINIT.COM) and wish to recover the data records in that file, you must use SENSRKEY.COM. Once the data file has been re-keyed, SENSIBLE SOLUTION will be able to use the filename.KS file to determine the number and location of records stored in the filename.MS file -- effectively recovering the data records "lost" during file initializing or restructuring. For a complete discussion on when to use Initialize, Rekey, and Restructure, see the section on Data Structures in this reference manual.

Operation:

SENSRKEY can be called from the operating system level or from the main menu. If it is executed from the operating system level, you can type out a command line:

```
d>SENSRKEY filename ###
```

where filename is the name of the data file (do not include the file extension) and, optionally, ### where you may specify the number of records you want to re-key. If you do not enter a number at the end of the command line, all of the records within the file will be rekeyed.

If you choose to initiate REKEY by selecting main menu 7, "Rekey A Data File", a prompt will appear on screen requesting the name of the data file that you wish to rekey. After you enter the file name you will be asked to either enter the number of records that you want to rekey or to press the [RETURN] key to re-key the entire file.

If you first initialized a file containing "live" data and you want to recover the data records, you must enter a number equal to or larger than the number of records in that file or simply press the [RETURN] key to rekey all of the records in the file. Remember that any time that you use SENSINIT.COM to initialize a data file, SENSINIT will display the actual number of records in your file. The re-key program will automatically rebuild the index-key file (filename.KS) while displaying the key values of all records as it does so. The program will stop when it finishes rekeying the number of records that you specified or comes to the end of the data file.

When SENSRKEY has completed the re-key operation, it will again ask you for the name of a data file to be re-keyed. Press the [ESC] (escape) key to return to the main menu.

If you have a large multi-user system you should refrain from using SENSRKEY while other users have access to the system. This is because the SENSRKEY program involves a great deal of disk accessing, particularly with a large data file containing many key fields. User response time may suffer. For very large data files on multi-user systems, re-keying is best left to quiet periods such as overnight.

One other multi-user consideration should be mentioned here. SENSRKEY can not be initiated if some other user is currently accessing the same data file. However, the corollary is also true. Once you have successfully initiated SENSRKEY, any other user attempting to access the same data file will be "locked out".

Compile A Source Code File

Overview:

SENSCOMP.COM is the SENSIBLE SOLUTION language compiler program. It is used to compile a SENSIBLE SOLUTION command file (program) into executable SENSIBLE SOLUTION pseudo code. SENSCOMP.COM will search for a file containing the file name you specify followed by a **.SRR** extension. SENSCOMP.COM will then compile the **.SRR** file into pseudo code and write the code into a new file named filename.RUN. You can execute SENSCOMP.COM from either the operating system level or from the main menu.

Operation:

From SENSIBLE SOLUTION main menu, select option 6, "Compile A Source Code File". A prompt will appear at the top of your screen requesting that you enter the name of the file (program) you wish to compile. Enter the drive location letter and then the file name. If the file you wish to compile (filename.SRR) resides on the drive you are currently logged on to, simply type a space and then the file name.

The compiler program (SENSCOMP.COM) will search for the command source file (filename.SRR) on the specified disk drive. An error message will be displayed if the file is not found. Press any key to clear the error message; the SENSIBLE SOLUTION main menu will return.

The SENSIBLE SOLUTION compiler goes through four phases:

-- Checking for Target Labels
-- Checking for Goto/Gosubs
-- Checking for Screens/Reporter formats
-- Writing out compiled command file

... and for each phase tells which line of the command file it is scanning. This running comment alerts the operator that the compiler is working. Note that the compiler may pause for several seconds when integrating a screen format file into the compiled command file. This is because each field reference in a screen or report must be located and linked.

If any errors are encountered during the compile, an appropriate message is displayed. All error messages are self-explanatory. (Examples: THERE ARE DUPLICATE SCREEN NAMES IN THIS PROGRAM. A SCREEN MAY APPEAR ONLY ONCE IN A PROGRAM and THE FIELD LENGTH WOULD EXCEED THE RIGHT MARGIN) A complete list of error messages and explanatory comments will be found in the Appendix of this manual. If your program will not compile, modify your command file using main menu selection 4, "Source Code Editor" and then re-compile. When compilation is completed, SENSCOMP.COM will return you to the main menu.

Operation:

SENSINIT can be executed from either the operating system level or the main menu. If it is executed from the operating system level, you can type out a command line:

```
d>SENSINIT filename
```

where filename is the name of the data file (no file extension is required). When SENSINIT is finished you will be returned to the system level.

SENSINIT works by checking the Data Dictionary to determine the file structure, field identification, key identification, and the current disk drive location of the data file. SENSINIT does this by looking at the information in the two Data Dictionary files, RECFLE.MS, and FLDFLE.MS. If the appropriate data files corresponding to the file name you specified do not exist on the disk, SENSINIT will create the two new files **filename.MS** and **filename.KS.**

If there are any active data records in the file being initialized, SENSINIT will display the message:

This file has # data records.
Do you REALLY want to initialize this file (Y/N)?

Consider your response carefully. When you INITIALIZE a data file containing data, the records stored in filename.MS will not be affected. However, the record counter, the total number of records in filename.KS, will be set to zero. Therefore, it will appear to any SENSIBLE SOLUTION program that the data file is empty. The record counter can be reset by using the REKEY program and entering the number of data records displayed here.

If you have changed the structure of the records in a data file, never INITIALIZE a data file containing data that you wish to maintain. The data stored in filename.MS will become corrupted and the former valid structure of the data will not be recoverable. Always use SENSRTC.COM, the data file restructuring utility, to convert existing data files to a new structure without losing the data.

If the data file being initialized contains any overlay fields (type O), SENSINIT will make a second alignment pass of these fields to correct their locations. A warning message will be displayed if an overlay field extends beyond the logical record length. A warning message will also be displayed if a field which contains the beginning of an overlay has been removed.

Any time that SENSINIT initializes a file containing a record number field (type R), it will always create a zero offset as this type of field requires no physical field size.

If you have a multi-user type system, you should be aware that SENSINIT can not be initiated if some other user is currently accessing the same Data Dictionary file. However, the corollary is also true. Once you have successfully initiated SENSINIT, any other user attempting to access the same Data Dictionary file will be "locked out".

[ESC] [C]   This option will change the command at the line marked by the line prompt ">".

When you select Insert ([ESC] [I]) or Change ([ESC] [C]) mode by pressing the appropriate keys, the following command options will be displayed:

```
 Enter  =  If  Go  Mount  Save rec  Delete rec  Clear
 Find  Print  Trap  Execute  ! remark  Lock  Unlock
```

Each of the options (activated by pressing the key that matches the bold print Capital character) is a command. Each command has a subset of functions. You will be asked questions about each of these functions. Your answer will determine how the command is to perform during program execution.

When you select [I] or [C], ten *****'s, denoting a field, will be displayed in the upper left-hand corner of your terminal. The field is used to give this particular command line a label. The command file branches to command labels and, if you anticipate that this command will be branched to from other places in your program, you should give it a label.

After each command that you insert or change, the ten *****'s will again appear in the upper left-hand corner waiting for you to enter the next command label or press [RETURN]. Again, the decision is yours as to labeling a command or not. You can always go back to a command and change it.

Pressing the escape key will exit the editor, save the edited SENSIBLE SOLUTION command file, and return you to the SENSIBLE SOLUTION main menu or the operating system prompt.

When you have finished editing your command file, SENSCMD.COM will create a disk file with the file name extension **.SRR**. Continue the program development process by compiling the source code file (filename.SRR) into a run-time file (filename.RUN) by using menu selection number 6, "Compile A Source Code File" (SENSCOMP.COM). Remember, though, before the program can be compiled, you must first have initialized all data files (filename.MS/.KS) referenced by your program.

Initialize A Data File

Overview:

Every data file accessed by a SENSIBLE SOLUTION program must first be created (initialized). SENSINIT.COM is the data file initialization program that creates the necessary data files, filename.MS and filename.KS, to store data handled by a SENSIBLE SOLUTION program.

After you have used a program and built up a data base, there are times when you may wish to alter the definitions of the fields with the data file. If you ever change a key field to a non-key field or vice versa but do not change the location of these fields within the record, then you must also use SENSINIT to notify the key file (filename.KS) that the key designation has been changed. After you have used SENSINIT under these circumstances, you must also use the rekey program, SENSRKEY.COM to recover your data records.

If you ever alter the structure of a data file in a way that would affect the location of fields within the record, you must use SENSRTC.COM, the restructure utility, to convert existing data files to the new structure without losing data. See the section on Data Structures in this reference manual for a complete discussion on when to use Initialize, Rekey, and Restructure.

SENSINIT.COM performs four different operations as it initializes the data files. First, SENSINIT.COM reads the specified field definitions in the Data Dictionary. These definitions are stored in FLDFLE.MS/.KS. Second, it calculates the location ("Offset") of each field within the record, and stores that information back into FLDFLE.MS/.KS. Third, it creates a new filename.MS if one does not exist, and it creates a filename.KS to store key field information. Finally, it sets the record counter in filename.KS (a reserved memory space containing a value equal to the number of "active" records in the file) to zero.

If you find that your selection was in error, simply press the [ESC] (escape) key to redisplay the options on your screen and resume work. Pressing [ESC] will always redisplay your control options menu. This allows you to make a selection by either entering a single character to activate a function or moving the highlight to the appropriate selection and pressing [RETURN]. For example, the command to move to the "Next page" may be activated by pressing [ESC] (which calls up the menu and indicates to the computer that you are ready to make a selection), and then pressing [N] (the key for the letter N on your keyboard.) Or, you may move the highlight to the appropriate selection, "Next page", and press [RETURN].

The status line will show you the name of the file being edited, the line number you are working on, the total lines in the program, and whether the Insert mode is **ON** or **OFF**.

If you have opened an existing file, the first 17 lines of code will be displayed and a right angle bracket character (**>**) will mark the particular line of code you can edit. By pressing your {UP ARROW] and [DOWN ARROW] keys you can scroll through the source code file (filename**.SRR)** a line at a time.

If the file named does not already exist, you will be asked if you want to create a new file. By responding Yes, the filename**.SRR** file will be created and you can begin using SENSCMD.COM to write a program in SENSIBLE SOLUTION source code. The editing options available to you through the Source Code Editor follow:

[ESC] [Q]	Quit the source program you are currently viewing. You have the option of leaving the Source Code Editor completely or of loading another file to edit.
[ESC] [P]	Using this option you may scroll the source code to the previous 8 lines. On the screen you will see the top line move to the middle of the screen, revealing the 8 lines that came before it. A right angle bracket character ">" will mark the new position of this line.
[ESC] [N]	Using this option you may scroll the source code to the next 8 lines . On the screen you will see the bottom line move to the middle of the screen, revealing the 8 lines that follow it. A right angle bracket character ">" will mark the new position of this line.
[ESC] [B]	This option finds the beginning command in the source

file. It sets the line prompt ">" at the first line of code.

[ESC] [E] This option finds the ending command in the source file. It sets the line prompt ">" at the last line of code.

[ESC] [D] This option will delete the line at the position marked by the line prompt ">".

[ESC] [M] Using this option you may mark the beginning or the end of a list of command lines in your program. Having thus defined a block of text, you may perform file/block editing operations (e.g., deleting a block, transferring a block to a new position in the program).

[ESC] [W] A marked block of command lines in your program is written to a temporary file. This action will not clear your block marks.

[ESC] [R] A block of command lines which you have sent to a holding file will be "read" (inserted) into the program you are editing. The inserted block will follow the line marked by the line prompt ">".

[ESC] [T] Use this option to transfer a marked block of command lines to a new position within your program. The inserted block will follow the line marked by the line prompt ">".

[ESC] [K] Use this option to delete a marked block of command lines from your program.

[ESC] [H] Use this option to print a listing of the source code file (.SRR).

[ESC] [F] This option will position the line prompt ">" at the line number, line label, field, or file name which you specify. The search for a line proceeds only from the position in the program at which you exercise this option You can not search backwards through a program. The last criteria that you specified is stored allowing you to repeat your search rapidly.

[ESC] [I] This option will insert a line after the position marked by the line prompt ">". Note that the insert is after the line being pointed at, not in front of it.

Source Code Editor

Overview:

SENSCMD.COM is the SENSIBLE SOLUTION Language source code editor. You must use SENSCMD.COM to "write" (create) a program source code file (filename.SRR). The source code file can then be compiled into SENSIBLE SOLUTION pseudo code (filename.RUN) by using main menu selection 6, SENSCOMP.COM. The compiled program can then be run by using the SENSIBLE SOLUTION executive program, SENSIBLE.COM (main menu selection 1). The following discussion will describe the operation of the source code editor. For a detailed description of each SENSIBLE SOLUTION language command utilized by the editor, please refer to the section on "Language Commands" in this reference manual.

Operation:

SENSCMD.COM provides 15 different commands for entering or modifying a program. SENSCMD can be executed from either the operating system level or the main menu. If it is executed from the operating system level, you can type out a command line:

```
d>SENSCMD filename
```

where filename is the name of the source code file (no file extension is required).

When SENSCMD.COM is executed from the main menu the following prompt line will be displayed at the top of your screen:

```
+-------------------------------------+
| Load command-file source for editing |   Quit
+-------------------------------------+

The SENSIBLE SOLUTION Command Source Editor  V2.0C
==========================================================================
```

Use the space bar to move the highlight block to the desired selection and then press [RETURN]. You will then be asked to enter the drive location and the name of a source file. If you want the source file you will be creating to be stored on the default system drive, simply press the space bar to advance the cursor to the file name portion of the prompt. Enter the name of the file and press the [RETURN] key.

The following display will appear on your terminal:

```
  Change line   Insert line   Delete line   Begin source   End source
  Previous page   Next page   Read block   Mark block    Write block
  Transfer block   delete blocK   Hard Copy   Find line    Quit
@:FILENAME.SRR       On line: XXXX    Tot lines: XXXX  Insert off
==========================================================================
```

The words **Change line** are highlighted with a reverse video block. By pressing your [SPACE] bar, the highlight will move to the next option available through the editor. The highlight indicates the option you wish to choose. Pressing the [SPACE BAR] will move the highlight to the right and pressing the [BACK SPACE] key will move the highlight to the left. When you have selected the option you want, press your [RETURN] key and the option will be activated.

Another way to activate an editing option is to press the character key indicated by the upper case character lodged in each option. For example, press [D] to select the "Delete line" function. Throughout this manual, brackets will be used to indicate that we are referring to a specific key on your keyboard.

answer "Y", the screen will temporarily clear and 11 prompts will appear requesting all of the information necessary to create a new field definition in the Data Dictionary. For a complete discussion on creating a new field definition, read the discussion of ENTFLE.RUN, "Data Dictionary Maintenance."

[ESC] [R] This option will remove a field window from the screen. Place the cursor over the first character prompt of the field window and then type [ESC] [R]. The field will not be removed from the Data Dictionary; it remains part of the record structure. You will see an error message if there was not a field where the cursor was positioned.

[ESC] [S] Move the cursor to the first character position of the field window and then use this option to display the definition of the field. Field name, File, Size, Col, Row, and Key will be displayed at the top of the screen. You will see an error message if there was not a field where the cursor was positioned.

[ESC] [F] If you wish to change a field definition while you are screen painting, use this option. You will be asked whether or not you wish to save the changes. If you answer "Y", the definition for that field will be updated in the Data Dictionary (FLDFLE.MS/.KS).

[ESC] [C] If you wish to change a file definition while you are screen painting, use this option.You will be asked whether or not you wish to save the changes. If you answer "Y", the definition for that file will be updated in the Data Dictionary (RECFLE.MS/.KS).

[ESC] [H] You may send a format description to a disk file (filename.LST) or to the printer with this option. The description will include the screen layout and a list of all field windows including the row and column position of the first character of each field window.

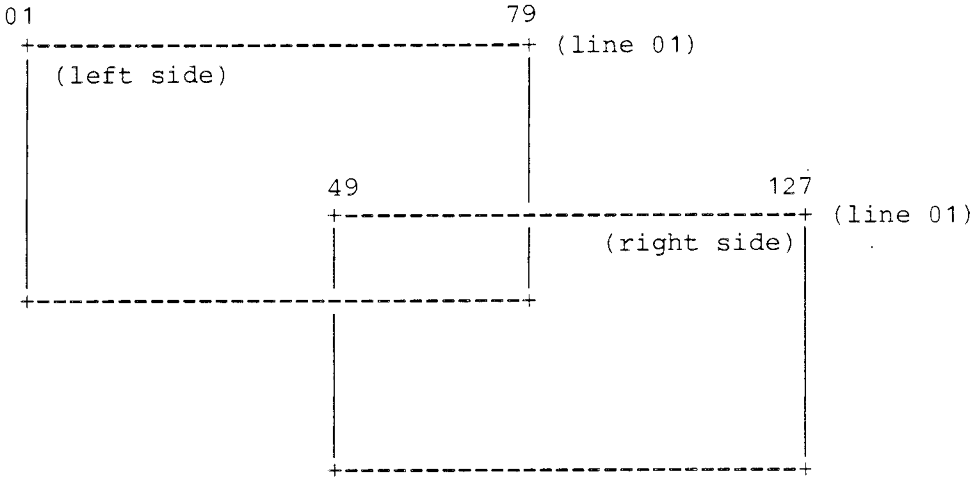
[ESC] [B] This option is used to create a box drawing on the screen. Place the cursor at the desired position for the top, left corner and type [ESC] [B]. Now move the cursor to the desired position for the lower, right corner of the box and press any key on your terminal. A box will appear on your screen. To create a horizontal line on your screen, simply create a box with no height. To create a vertical line on your screen, simply create a box with no width.

[ESC] [U] This option is used to remove box drawings from the screen format. Place the cursor at the top, left corner of the box you wish to remove and type [ESC] [U].

[ESC] [E] As you screen paint, use this option to redisplay the screen and redraw boxes that have been altered due to insertions and deletions.

When you have typed screen messages and placed fields as desired, press [ESC] [Q] to exit Screen Painting. SENSCRN.COM will ask you "Save this screen file? (Y/N) Y". **Yes** is the default answer.

If you choose not to save the screen, SENSCRN.COM will ask: "Abandon this screen file? (Y/N) N". A "N" answer (the default) will re-display the screen as it was and allow you to continue editing.



Use [ESC] [J] To Shift Views

As we stated earlier, you can create report formats up to 255 columns wide. To extend a report format line beyond the 127th. column, simply place a comma in the 127th. column. The next line down on the format screen becomes columns 127 through 255 and will be attached to that line so that when the report is sent to the printer, the two different lines on your screen will be printed on the same line level on paper.

You can also create up to 60 format lines on your screen. Use the down arrow key to advance to the next line and your screen display will scroll down one line at a time.

Screen Painting Control Keys

[Up arrow]	Pressing this key moves the cursor up one line.
[Down arrow]	Pressing this key moves the cursor down one line.

[Right arrow] Pressing this key moves the cursor one space to the right.

[Left arrow] Pressing this key moves the cursor one space to the left.

[CTRL] [I] A blank is inserted under the cursor. All characters to the right of the cursor shift right and the last character on the line is lost.

[CTRL] [U] Whatever is in the field window will be deleted and the cursor will move to the left side to await another entry.

[CTRL] [D] The character under the cursor will be deleted. All characters to the right of the cursor shift left.

[ESC] [J] SENSCRN.COM will allow you to paint both screen formats and report formats. The width of a report format is limited to 255 columns. Pressing [ESC] [J] will shift your display to the right or left so that you may view 127 columns of a report format with your 80 column CRT.

[ESC] [D] The line below the cursor is deleted. All lines below move up and a blank line is created at the bottom of the screen.

[ESC] [L] A blank line is inserted below the cursor. All lines below move down and the last line on the screen is lost.

[ESC] [Q] This ends an editing session. SENSCRN.COM will ask: Do you wish to save the changes? Answer with a "Y", "N", or [RETURN]. The screen painting program will then return to the original prompt line at the top of the screen that requests the type of format you wish to create.

[ESC] [A] The screen painting program will ask for the name of the field to be placed. The first character in the field will be at the current cursor position. Field names may be up to 15 characters long. If the field is already in the Data Dictionary it will be placed automatically. If the SENSCRN.COM can not find the field in the Data Dictionary, a message will be displayed indicating that the field can not be found. It will continue to ask if you want to create a new field definition. If you

Screen Painting

Overview:

This program will allow you to create and define a SENSIBLE SOLUTION screen format or a printer report format and then store that format in a file named filename.**SCC**. Screen formats or "templates" are accessed by SENSIBLE SOLUTION programs to allow data entry and retrieval from your terminal screen display. Reporter formats are used primarily to define the layout of a report that will be sent to the printer device.

Operation:

SENSCRN.COM can be called directly from the operating system mode or it may be selected from the main menu. There are two phases to SENSCRN.COM operation, screen painting (definition) and field definition and placement.

When SENSCRN.COM is executed, your display screen will immediately go blank and the following prompt line will be displayed at the top of your screen:

```
                                         +---------------+
Enter the type of format you wish to load | Screen Format | Reporter Format
                                         +---------------+
```

Use the space bar to move the highlight block to the desired selection and then press [RETURN]. You will then be asked to enter the drive location and the name of a format. If you want the format file you will be creating to be stored on the default system drive, simply press the space bar to advance the cursor to the file name portion of the prompt. Enter the name of the file in which you want the format to be stored and press the [RETURN] key. The following line will be displayed at the top of your screen:

```
        @:filename.SCC file opened  col = 001  row = 01
```

Screen Format

The SENSIBLE SOLUTION Screen Painting program (SENSCRN.COM) allows you to interactively define a screen format for use with SENSIBLE SOLUTION programs. Because screens are often used for simple data entry and update, the SENSIBLE SOLUTION also provides a facility to automatically generate a file maintenance program. See Main Menu Selection 9 -- "Program Generator."

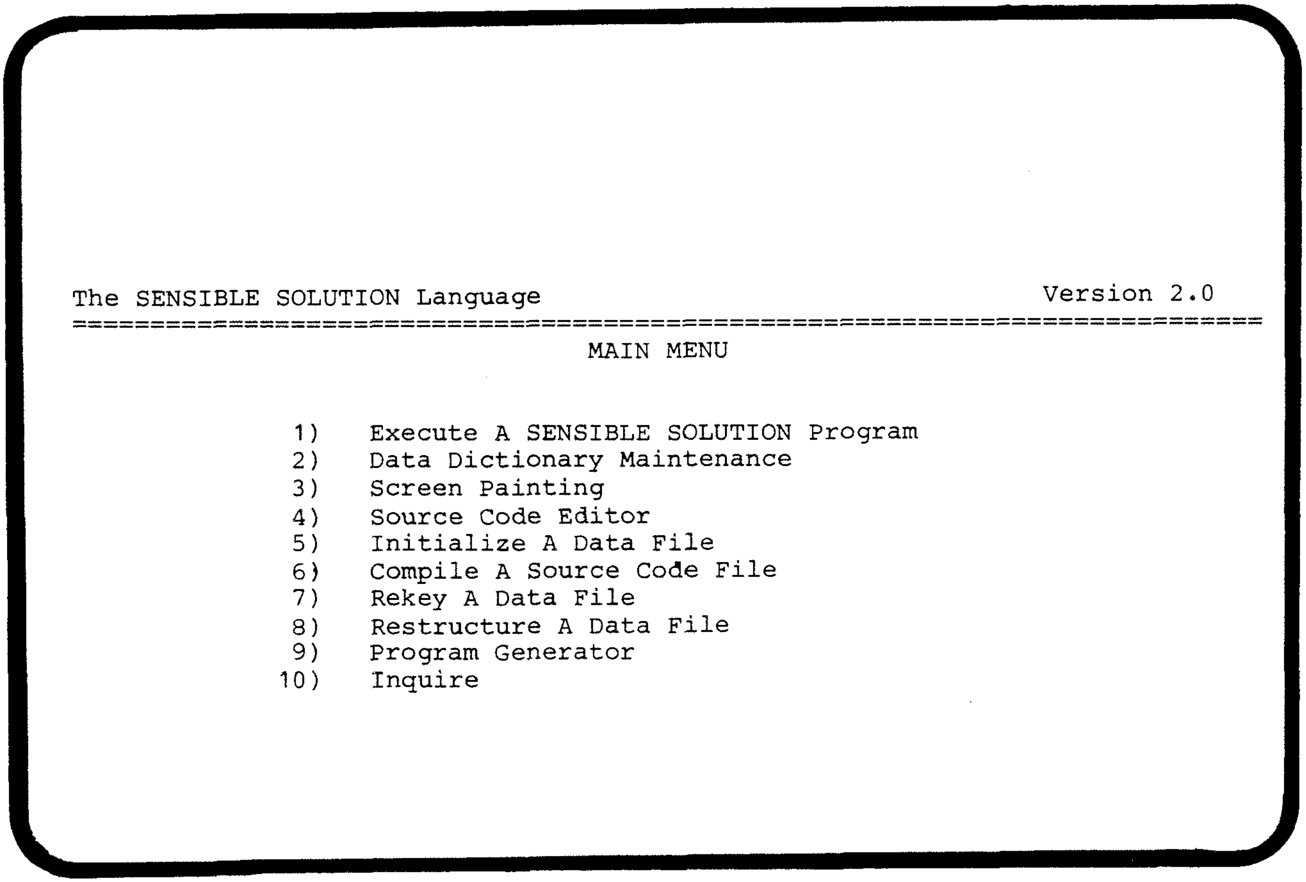
A screen format can be a maximum of 79 columns in width and 22 lines in height. Use the arrow keys on your keyboard to direct the cursor to any position you want on the screen. You may then use the various escape commands and control keys described on the next page to "paint" messages, titles, box drawings, horizontal, and vertical lines. You may also place field "windows" on the screen where data entry and retrieval will take place.

Reporter Format

The Screen Painting facility, SENSCRN.COM, will also allow you to create report formats that can be utilized by SENSIBLE SOLUTION programs to send formatted reports to your computer's printer. Reporter formats differ slightly from screen formats.

For one thing, you can not use the box drawing feature that is available with screen format. Another difference is that a reporter format can be much larger than a screen format: a maximum of 255 columns wide and 60 lines in height. Obviously this is considerably larger than a standard display screen of 80 columns by 24 lines. The screen painting facility, however, has a method for getting around the limitation of your screen size.

The screen painting escape command, [ESC] [J], will allow you to shift back and forth between two views of the reporter screen. The left screen is defined as column 01 through column 79; the right screen is defined as column 49 through column 127. Thus, there is a 30 column overlap area from columns 49 through 79. This overlap area will make it easy for you to "keep your bearings" as you shift back and forth between the two areas of the screen.



```
The SENSIBLE SOLUTION Language                                   Version 2.0
============================================================================
                                 MAIN MENU


                  1)    Execute A SENSIBLE SOLUTION Program
                  2)    Data Dictionary Maintenance
                  3)    Screen Painting
                  4)    Source Code Editor
                  5)    Initialize A Data File
                  6)    Compile A Source Code File
                  7)    Rekey A Data File
                  8)    Restructure A Data File
                  9)    Program Generator
                 10)    Inquire
```

Executing A SENSIBLE SOLUTION Program From The Operating System Level

To execute a compiled SENSIBLE SOLUTION program from the operating system level type:

```
     d> SENSIBLE  d:<filename>[RETURN]
          (where "d" is the disk drive location letter and [RETURN] is the
          carriage return key)
```

For example, if you are logged onto drive "A:" and you would like to execute the SENSIBLE SOLUTION Management Series program "MGLCHAR.RUN", which is on drive "E:", simply type the following command line from the operating system level:

```
A> SENSIBLE E:MGLCHAR[RETURN]
```

If the SENSIBLE SOLUTION program you wish to execute does not reside on the default drive (the logged on drive), you will have to specify the drive location of the compiled program, <filename.**RUN**> as shown in the example above.

Executing A SENSIBLE SOLUTION Program From The Main Menu

From the main menu level, select option number 1, "Execute A SENSIBLE SOLUTION Program." A prompt will be displayed on your screen asking you for the drive location letter and the name of the file (filename.RUN) that you want to execute. If the program resides on the currently logged on drive, press the space bar to advance the cursor into the file name portion of the prompt. Enter the name of the program (filename) and the program will be executed. If the program name (filename) is shorter than 8 characters, you will have to press the [RETURN] key to begin execution. If the program (filename.RUN) cannot be found on the designated disk drive, SENSIBLE.COM will report an error message.

If the file (program) is found, control will immediately pass to the specified SENSIBLE SOLUTION program. The program may specify a transfer to another SENSIBLE SOLUTION program. When the program is finished, control will return to the language main menu.

Several different key stroke controls are available to control the execution of a SENSIBLE SOLUTION program. These keys and their actions are as follows:

SENSIBLE SOLUTION Execution Control Keys

[CTRL] [I] Inserts a blank "under the cursor". Remaining characters in field shift right. Rightmost character in field is lost.

[CTRL] [U] Clears the displayed value from the field window.

[CTRL] [D] Deletes the character "under the cursor". Remaining characters in field field shift left to fill, and a blank appears at right end of field.

[LEFT ARROW] Move cursor one position left within field. Value of the field is unchanged.

[RIGHT ARROW] Move cursor one position right within field. Value of the field is unchanged.

[UP ARROW] Move cursor to beginning of previous field. (Locks at topmost field)

[DOWN ARROW] Move cursor to beginning of next field. (Locks at lowest field)

[ESC] [J] Display next screen.

[ESC] [Q] Display previous screen.

[ESC] [S] Save record.

[ESC] [R] Delete record.

[ESC] [F] Finds the record, which contains the field value, that most closely matches the displayed field value. Possible errors: "not a key field", "end of file encountered (record not found)"

[ESC] [B] Finds first record (lowest value) in the file based on the field in which cursor appears.

[ESC] [E] Finds last record (highest value) in the file based on field in which cursor appears.

[ESC] [N] Finds next record in file. Possible error: "end of file encountered."

[ESC] [P] Finds previous record. Possible error: "beginning of file encountered."

[ESC] [C] Clears all fields on screen to spaces and clears the buffer.

[ESC] [?] Display help screen.

NOTE: "Find next" and "find previous" must be preceded by a "find", "find beginning," or "find ending" in the same field. All searches trigger the Relates Trap command.

Trace -- The SENSIBLE SOLUTION program debugger

SENSIBLE.COM has a program debugger mode built into it called "Trace". This Trace function will allow you to execute your SENSIBLE SOLUTION programs one line at a time, display values, change values and continue processing, and set traps to stop at specified places in the program. These features will provide you the programmer with a sophisticated environment for debugging your programs.

Before the Trace program debugger can be used, you must first have enabled it when SENSIBLE SOLUTION was originally installed on your computer system. Refer to the section on SENSETUP.COM in the Installation Manual.

Once the Trace mode has been enabled on your system, you can invoke it at any time during the execution of a program. Press **[ESC] [T]** and the following seven options will be displayed at the bottom of your screen:

```
Line #, [ENTER] continue, [SPACE] single-step, N perform (n) lines
T trap input,   F field inspect,   L stop at line,   G goto line
```

Here is an explanation of each one of the Trace command options:

Line #:
This area at the bottom of your screen will display the current program line that is being executed.

[ENTER] to continue:
If you press the [ENTER] key, you will exit the Trace mode. You can do this at any time.

[SPACE] single-step:
Press your space bar to execute the current line of the program. After the line is executed the seven trace options will again be displayed at the bottom of your screen along with the next program line number. This may cause you some confusion if the next line in the program happens to be an ENTER command. If this is the case, you will have to enter a value into the current field window before you can continue back into single-line trace mode.

N perform (n) lines:
This Trace option will allow you to execute (n) number of lines from the currently displayed Line #. You can execute a maximum of 255 lines in N trace mode. After the system has executed the specified number of lines, the 7 trace options will again be displayed at the bottom of your screen. The Line # value will display the command line number where execution stopped.

T trap inspect:
This option will allow the program to continuously execute until it encounters one of the Trap options you wrote in the program. When the Trap is encountered the program will immediately switch to Trace mode and the 7 trace options will be displayed at the bottom of your screen. The displayed Line # will be the program command line that encountered the Trap condition.

When you select "T" for trap inspect, the following display will appear at the bottom of your screen:

```
Save  Delete   Clear  eXit     File-err  Related  Jumpscreen
Help   Locking  Page  Uparrow  dowNarrow     (press one key)
```

Indicate the Trap option that you want and the system will display:

```
<current status>                 Change target line?  (Y or N)
```

The area marked as <current status> will display either "default", <a number ###>, or "ignore". "default" means that the standard system default is set for the Trap. "ignore" means that the Trap has been set to be ignored. If a number is displayed it represents the program line number where program flow will branch upon encountering the specified Trap. If you choose to change any one of these three possible <current status>, press "Y" and the following message will be displayed:

```
**********    Enter target line# or label  (0=default, 65535=OFF)
```

This prompt will allow you to enter either a line number or a line label where program flow will branch when the Trap is encountered. If you wish, you can set the Trap to system default by entering a "0", or you can enter "65535" to turn off the trap during program execution. This feature will allow you to continuously execute programs in the normal mode and automatically invoke the Trace mode when the specified conditions occur.

F field inspect:
This Trace option will allow you to view or change the value in any field at any time during program execution. To activate this Trace feature, press "F" and the following display will appear:

********************** Fieldname (blank=done)**

Enter the name of the field that you want to inspect or alter. The following prompt will be displayed:

<fieldname> ## Array entry # ([CR]= not array)

If the field is an array, enter the element number that you wish to display. If it is not an array, press the [RETURN] key. The value currently in the field will be displayed. You may change this value; however, if you change it, this Trace feature will update the field with the new value. To return to single line execution mode, clear the <fieldname> window with [CTRL] [U] and press [RETURN]; your screen will display the 7 Trace options.

Before returning to execution mode you may be asked:

Trap on assignment? Y/N

If you respond "Y", any time a value is moved into this field, the system will automatically invoke the Trace mode and stop execution. This feature will allow you to continuously execute programs in normal mode and then automatically invoke the Trace mode when the proper set of conditions occur.

L stop at line number:
This Trace feature will allow you to set a line number or label to designate where you want to halt program execution and automatically invoke the Trace mode:

****************** Line# or label to trap (0=off)**

Enter the line number of the label where you want program execution to halt and Trace mode to begin. When the specified line number or label is reached in the program, Trace mode will be automatically invoked. At that point, you can display fields, change values in fields, etc.. To turn off this automatic Trace enable feature, enter a "0" in the field window.

G goto line:

The Trace option described above lets you automatically turn on the Trace mode. The "G" option described here will let you specify where you want Trace mode to automatically turn off. Follow the same procedure as that outlined above under "L", but instead, specify the line number or label where you want the Trace mode turned off and normal program execution to continue.

Data Dictionary Maintenance

Overview:

The structure of every file and field accessed by a SENSIBLE SOLUTION program is defined in a data dictionary. The Data Dictionary contains all of the information on file and field names, file disk drive location, data types, field lengths, field masks, field offsets (field position within a file record), keys, and comments on file usage. ENTFLE.RUN is a SENSIBLE SOLUTION program that creates and maintains the Data Dictionary.

Before you can compile and execute a SENSIBLE SOLUTION program, you must first define in the Data Dictionary all fields and files that are going to be accessed by that program. In fact, every time a SENSIBLE SOLUTION program is run it must access the Data Dictionary to determine the location of the proper data files. However, the Data Dictionary does not create or re-build data files; it only defines the structure of the data files.

Operation:

Since ENTFLE.RUN is a SENSIBLE SOLUTION program, it uses the same screen controls to enter data into the dictionary and retrieve data from the dictionary as any other SENSIBLE SOLUTION program. If you are not yet familiar with the screen controls used to execute a SENSIBLE SOLUTION program, refer to the section on SENSIBLE.COM.

ENTFLE.RUN maintains 4 data files, **RECFLE.MS/.KS**, which contains information about the files and keys, and **FLDFLE.MS/.KS**, which contains information about fields.

ENTFLE.RUN can be executed directly from the operating system by typing the command line:

```
d>SENSIBLE ENTFLE[RETURN]
(where "d>" is the currently logged on disk drive)
```

or you may select main menu selection number 2, "Data Dictionary Maintenance."

When ENTFLE.RUN is executed, a template will appear on your screen that will allow you to create, modify, and delete files and fields from the Data Dictionary. The upper portion of the template will allow you to select and modify data file definitions and the lower portion of the screen will allow you to select and modify data field definitions. ENTFLE.RUN will also allow you to print out a report of all field definitions for a particular data file.

When you have finished using ENTFLE.RUN to define the files and fields that will be accessed by your new program, you must then INITIALIZE the file by using the module SENSINIT.COM (main menu selection 5). When you use ENTFLE.RUN to change the structure of an existing data file, you must also RESTRUCTURE the file and possibly REKEY it as well.

The Structure Of The Data Dictionary

A SENSIBLE SOLUTION data file consists of two disk files, filename.**MS** and filename.**KS**. Filename.MS is the master data file, which contains the actual data stored in straight ASCII character code. The location of each field within the record is determined from field definition information stored in FLDFLE.MS. Fields are not delimited by commas or other field separators. Records are stored on 128-byte block boundaries. For example, a 500-byte record occupies four 128-byte blocks (total 512 bytes) with 12 bytes ignored at the end of the fourth block. This speeds disk access and insures correct record locking on multi-user operating systems.

Filename.**KS** is the index-key file, which maintains a "B*-tree" of record pointers in the master data file (filename.MS). This method provides exceptionally fast access to data records. Although all data files generated by SENSIBLE SOLUTION are in straight ASCII format, the B*-tree format is not in straight ASCII.

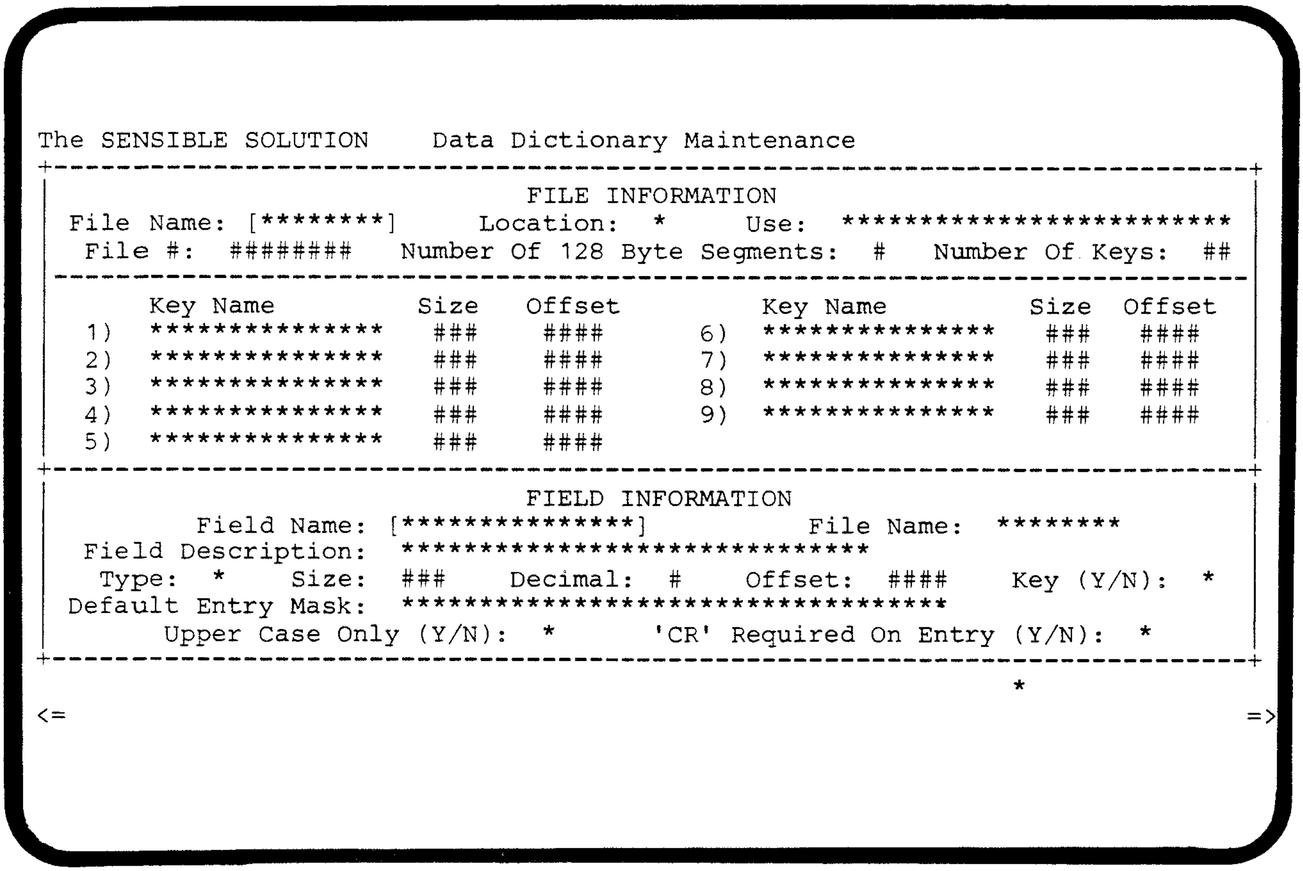
The SENSIBLE SOLUTION Language also includes utilities for data file maintenance: initialize a data file (SENSINIT.COM), re-key a data file (SENSRKEY.COM), and restructure a data file (SENSRSTC.COM). These utilities are explained in detail later.

Using Data Dictionary Maintenance

You can execute the SENSIBLE SOLUTION program ENTFLE.RUN from the operating system level. Type:

```
d>SENSIBLE ENTFLE[RETURN]
(where "d" is the operating system drive location prompt)
```

or select option number 2 from the language main menu, "Data Dictionary Maintenance." The following template will appear on your screen:



```
The SENSIBLE SOLUTION    Data Dictionary Maintenance
+----------------------------------------------------------------------------+
|                            FILE INFORMATION                                |
| File Name: [********]    Location:  *    Use:  ************************* |
|  File #:  ########   Number Of 128 Byte Segments:  #   Number Of Keys:  ## |
 ----------------------------------------------------------------------------
|      Key Name           Size   Offset          Key Name          Size  Offset |
|   1)  ***************   ###    ####       6)  ***************   ###   ####  |
|   2)  ***************   ###    ####       7)  ***************   ###   ####  |
|   3)  ***************   ###    ####       8)  ***************   ###   ####  |
|   4)  ***************   ###    ####       9)  ***************   ###   ####  |
|   5)  ***************   ###    ####                                         |
+----------------------------------------------------------------------------+
|                            FIELD INFORMATION                               |
|         Field Name: [***************]         File Name:  ********          |
|   Field Description:  ******************************                       |
|    Type:  *    Size:  ###    Decimal:  #    Offset:  ####    Key (Y/N):  *  |
|  Default Entry Mask:  ***********************************                 |
|        Upper Case Only (Y/N):  *      'CR' Required On Entry (Y/N):  *     |
+----------------------------------------------------------------------------+
                                                                  *
<=                                                                          =>
```

This screen will update both RECFLE (the file definitions) and FLDFLE (the field definitions). You may create a new definition or modify an existing definition by typing the value in the "File Name" field window and then pressing **[RETURN]**. You can also search for an existing value by typing the (partial or full) key value, then using [ESC] [F] to find the closest approximation of the value or by using the find beginning/ending/first/last controls (see SENSIBLE.COM). A new or modified definition is saved in the Data Dictionary by pressing [ESC] [S]. You may also delete definitions from the Data Dictionary by pressing [ESC] [R].

The Data Dictionary screen is divided into two parts: FILE INFORMATION, and FIELD INFORMATION. The following is a brief discussion of each field window on the Data Dictionary screen:

FILE INFORMATION

File Name:
Enter the name of the file that you wish to create, delete, or edit. The file name can be no longer than 8 characters in length.

Do not change the "File Name" of an existing data file unless you intend to change the name of the actual data file on the disk and all references to the data file in your programs.

If you delete a file name by pressing **[ESC] [R]**, the entire file definition will be deleted including all of that file's field definitions.

File Number:
The value displayed in this field window represents the internal file number maintained by ENTFLE.RUN. You can not directly change the value.

Location:
This field window will allow you to specify the disk drive location of this particular data file (filename.MS/.KS). If you are creating a new file definition, this field will automatically default to the currently logged on drive location. If you save a file definition and the "Location" field window is left blank, the Data Dictionary will assume that the new data file is located on the currently logged-on drive.

Number Of 128 Byte Segments:
This field window will display the currently defined size of a record in this particular data file (filename.MS). The displayed value is expressed

in 128 byte units.

Use:
You may write a comment on the use of this particular file.

Number Of Keys:
This field window will display the number of indexing keys used by this file.

Key Name:
The Data Dictionary will allow you to specify up to 9 different key fields. A key field name can not exceed 15 characters in length. A new data file must be INITIALIZED (see SENSINIT.COM, Main Menu Selection number 5) before any of the "Key Name", "Size", or "Offset" information will be displayed in this portion of the Data Dictionary screen. Thus, if you change a key field definition or create a new key field in the FIELD INFORMATION portion of this screen, that change will not appear in any of these three fields until you have INITIALIZED the data file.

Size:
These field windows will display the size of each key field specified for this particular file. The size of the key field is expressed in bytes. A key field can not exceed 72 characters in length if it is an alphabetic type field (type "A") and 14 digits in length if it is a numeric type field (type "N").

Offset:
These field windows will display values (in bytes) that indicate the location of the key fields within the record. The displayed values represent the offset of the first byte in each key field.

FIELD INFORMATION

To move the cursor into this portion of the Data Dictionary screen you must use the "jump" command -- [ESC] [J]. To return the cursor to the FILE INFORMATION portion of the screen use the "quit" command -- [ESC] [Q].

Field Name:
This field window will allow you to create, edit, or delete a field definition by referring to the name of the field. The field name you enter here must not exceed 15 characters in length. The field name may contain any alphanumeric characters and some punctuation marks. You can not use any of the following characters in a field name:

( ) [ ] < > / + - = * &

The brackets surrounding the field window on this screen indicate that this field is a key field. You can, therefore, conduct a search for any "Field Name" value currently on file in the Data Dictionary by using the appropriate SENSIBLE SOLUTION screen controls (see SENSIBLE.COM).

File Name:
The file name displayed in this field window is the same file name displayed in the FILE INFORMATION block on the screen. It is a key field and you can conduct a search for any file name currently on file in the Data Dictionary.

One important point should be made here concerning the relationship between "Field Name" and "File Name" when you are conducting a search. As you scroll through the field names currently on file, the "File Name" field window will always display the correct file name for that field name. File names are arranged in alphabetical order and there corresponding field names are arranged in alphabetical order under their respective parental file names:

Field Description:
This 30 character field window will allow you to enter a description of the field you are defining.

Type:
You may specify the type of field that you are creating. The SENSIBLE SOLUTION will allow you to use 5 different types of fields. Every field in a file must be defined as to its "type":

Type	Data
A	alphanumeric characters
N	numeric data
D	date entered in Gregorian form and stored in Julian form
O	overlay field, stored as an alphanumeric string
R	record number

Type "A" field is strictly alphabetic characters and numbers. This type of field is generally used for strings. You can not use numbers written in type "A" fields as numbers for performing calculations.

Type "N" field is composed of numbers 0 through 9. Any fields that are involved in numeric calculations must be designated as type "N".

Type "D" field is strictly dates. Dates are entered into SENSIBLE SOLUTION programs in one of several forms -- mm/dd/yy, dd/mm/yy, mm/dd/ccyy, or dd/mm/ccyy. The date format that your system uses will depend upon the date format that was specified during the initial setup -- see SENSETUP.COM. All dates are stored internally in SENSIBLE SOLUTION programs as a Julian Date number. The number is equal to the number of days from January 1st. 0000 A.D. to the specified date.

Type "O" fields are overlay fields. These fields are composed of two or more fields. An overlay field will always interpret the fields it is subtending as alphanumeric string type fields.

Type "R" fields are record fields. A record field is used to temporarily store a value to locate and retrieve any desired record from a file.

There is a thorough discussion of these five different types of fields in the section on Data Structures in this reference manual.

Size:
You must enter the size or "width" of the field you are creating or editing in this field window. Most fields can be a maximum of 255 characters in length. Key fields, however, can only be a maximum of 72 characters in length and fields that contain numeric type data (type "N") can only be 14 characters in length.

Decimal:
Enter the number of decimal places that will be recognized by this field definition. You can not exceed 4 decimal places.

Offset:
Most fields that you create will not require an offset value. Consequently the cursor will rarely move into this window. There is one exception, however. Overlay type fields (type "O") will require that you specify an offset value in this field window. The offset value is the number of bytes from the start of the record to the beginning of the field. We will discuss the use of this offset in more detail under the section on overlays.

If the field you are currently defining is not a type "O" field and there is a value displayed in the "Offset" field window, that value was calculated by the system at the time the data file was initialized.

Key (Y/N):
You must enter a "Y" or "N" into this field window to indicate whether or not the field you are defining is a key field.

Default Entry Mask:
This 35 character width field will allow you to enter "mask" parameters that will define the type of data that can be entered by the computer operator. Later on in this discussion we will give a thorough description of the mask parameters available to you and show you the real power of this masking feature.

Upper Case Only (Y/N):
This field window will allow you to define the type of characters (upper case or lower case) that can be entered by the computer operator.

'CR' Required On Entry (Y/N):
This field window will allow you to define whether or not the computer operator must press the [RETURN] key after he or she has filled a field window. For example, if you have defined a 5 character zip code field and have answered "Y" to this question, the computer operator will have to press [RETURN] after entering the fifth digit in order for the program to continue execution.

Defining An Input Mask For A Field

The SENSIBLE SOLUTION will allow you to define an "input mask" for a field. An input mask defines the manner in which a field will be entered from the keyboard when an ENTER program command is executed. The input masking techniques used will allow left or right justification of fields within the field width, automatic formatting, "forced" character entry, and character-by-character validation against "inclusion" or "exclusion" classes.

As you recall from the previous discussion, you can define the type of data that a particular field window will handle such as:

*****	accepts any keyboard character except control characters
A	accepts any alphabetic character or space
#	accepts any digit, 0-9 (for NUMERIC fields, "#" will also accept leading spaces or a minus sign, when appropriate)

An input mask definition essentially controls how this particular type of field can be entered by the computer operator. The mask itself is a string that is stored in the Data Dictionary in the field window called "Default Entry Mask." There are 9 different characters that can be used in the input mask string to define how each operator keystroke will be accepted and handled by the program:

[ ] ( ) < > ! ? \

You may define certain classes of characters that will be either accepted or rejected at a given position in a field window by using square brackets in the "Default Entry Mask" field window. Square brackets that face in [ ] indicate that the characters inside the brackets will be accepted by the field window. Square brackets that face out] [indicate that the characters inside the brackets will not be accepted by the field window:

[AEFHX..Z] — accepts any of the characters A, E, F, H, X, Y, or Z. "X through Z" is represented by X..Z. If the field has been marked UPPER CASE in the Data Dictionary, lower case letters will be converted to upper case before testing for acceptance.)

]13579[— will accept any printing character EXCEPT the odd digits that is, any even digit, space, letters or punctuation.

You may also indicate the characters that will be accepted at a single position and/or adjacent positions within the field window by using parentheses in the input mask string:

(10)A — accepts ten alpha (or space) characters

(2)[YN] — accepts two "Y" or "N" strokes, but no other characters.

The left and right angle brackets can be used to indicate how each character is displayed in the field window after each keystroke:

> — indicates that keystrokes accepted into the field will appear under the cursor and the cursor will then move to the right.

< — indicates that keystrokes accepted into the field will

appear at the right side of the field window and then will move leftward as other characters are entered -- much the same way that a pocket calculator display works.

You can even mix these last two input mask characters to achieve a sophisticated entry display. For example, if you specified the following input mask:

<#######>.###

the operator's screen would first accept up to 7 digits in the "units" part of a number from right to left. When the units part is full, OR the operator types a period, the field window will then accept up to three decimal digit key strokes and display them from left to right.

You may specify whether or not an entry by the computer operator is absolutely required by using an exclamation point in the input mask string:

! indicates that characters after the "!" MUST be entered into the field. The input routine will not permit you to move the cursor out of the field window until these characters have been filled with keystrokes.

You may specify that a field window entry be optional by using a question mark within the input mask string:

? indicates that the character positions after the "?" may be left unfilled. Under this condition an ENTER program command will automatically fill in "empty" positions with a space or some other appropriate character. In the case of numeric type fields, for example, the field window would be filled with trailing zeros. The "?" character in an input mask is the regular default value of an input mask. That is, character positions may normally be left unfilled in a field window.

Both of the input mask characters "!" and "?" may be mixed within a field window input mask definition. For example:

!###/?###-#### accepts a telephone number like 206/555-1212. The area code is required but the last 7 digits are optional; an ENTER program command will not permit the user to exit the field until the first 3 digits have been entered.

You may use the backslash character "\" to force the following character in a mask to be taken literally. This option will allow the operator to enter characters that are identical to the input mask special characters like parentheses. Non-masking characters do not need to be forced with a backslash character. This lets you format a field without the need for special parsing or computation. For example:

\ backslash indicates that the next character in the mask is to be taken literally as a "forced" character.

\(<###\) >!###-#### is a mask that allows entry of a telephone number in the form **"(123) 456-7890."** The parentheses and dash are "forced" and appear automatically. The area code is optional but the last 7 digits of the number must be entered. An ENTER statement in a program will not let the operator leave the field window until they are typed.

Part \#A-<##### accepts a part number such as "Part #A-   78". The "Part #" is "forced" and appears automatically. The digits portion of the field is right-justified within the last 5 positions of the field to sort correctly on its numeric value.

Field Listing For File

Once the cursor is residing in the lower portion of the Data Dictionary screen, FIELD INFORMATION, you can then jump to another screen by pressing [ESC] [J]. The second screen of ENTFLE.RUN will allow you to print out a a report of all field definitions associated with a specified file. Simply enter the data-file name and press the [RETURN] key. A message will appear at the bottom of your screen requesting the destination of the report -- the printer, the CRT, or the disk. Choose one of the three options and a 116 column report will be produced. The report will include all of the field information from the Data Dictionary screen for the particular file that you specify:

Field Name
Field Description
Type
Size
Decimal
Key (Y/N)
Offset
Upper Case Only
'CR' Required On Entry (Y/N)
Default Entry Mask

To return the cursor to the FIELD INFORMATION portion of the Data Dictionary screen press [ESC] [Q].

Special Considerations When Using The Data Dictionary

There are a few special considerations concerning the Data Dictionary of which you should always be aware:

-- Deleted files should be erased from the disk. DO NOT ERASE A FILE IF IT IS STILL IN THE DATA DICTIONARY.

-- Location-changed files must be moved to their new drive. Do not forget to move both the FILENAME.**MS** and FILENAME.**KS** files.

-- Newly-defined files must be INITIALIZED before a program can be compiled. INITIALIZING the data file sets all of the field "offsets" (defines the location of the fields within the record) for the Data Dictionary.

-- Re-defined files must be RESTRUCTURED to accommodate the changes in field definitions. If the fields have not changed in length or type, but the key-fields have been changed, you must INITIALIZE and then REKEY the file.

-- If you have modified the length or the mask definition of any field, you must use menu selection 3, "Screen Painting" to re-load the file into SENSCRN.COM (screen painting module) and then re-save the screen format. All SENSIBLE SOLUTION programs that access the modified data files must be re-compiled since the compiler stores information about the data structure in the compiled program. If you have changed the file names, you must first modify the source code before compiling.

For more information concerning the Data Dictionary refer to the fourth section of this reference manual, which covers data structures.

LANGUAGE COMMANDS

Table of Contents

Enter

PURPOSE:

This command is used to enter data from the terminal or from a record that has been read from the disk.

USAGE:

Select ENTER by pressing **[E]** or pressing **[RETURN]** when the option is highlighted. After the selection you will see the word "enter" followed by a prompt of 15 asterisks displayed on your screen.

```
enter ***************
```

Type the name of the field that is to receive the data. The EDITOR will consult the Data Dictionary to verify that the field you named exists. If it does not, a message to that affect will be displayed on your screen, and the EDITOR will wait for you to enter an existing field name.

After entering the field name, you have the option to define a mask for the field. This mask defines the manner in which data will be entered from the keyboard when an enter command is executed. The masking techniques used will allow left or right entry of data within the field width, as well as automatic formatting, "forced" character entry, and character-by-character validation against "inclusion" or "exclusion" classes. (Masking is explained in the discussion of the Data Dictionary in the section on "Data Structures.") The default standards are right to left data entry for numeric fields and left to right data entry for alphanumeric fields.

A 32 character wide prompt will be displayed on your screen that will contain the default entry mask as defined in the Data Dictionary. If the prompt line is empty then no entry mask was previously specified in the Data Dictionary definition for that field. In either case, you may elect to override the default mask for this one particular user entry by simply specifying a new entry mask. Your new definition will apply for this entry only; it will not change the Data Dictionary definition of this field throughout your program.

Next the EDITOR will ask you if you wish to designate this field as a password. If you answer "yes", data entered into this field from the keyboard will not be displayed on the screen.

If you want the program to branch to a help screen for the field you have named, you can give the label of the command line in your program that displays the help screen. Subsequently, this help screen will always appear when the **[?]** key is pressed. If you have multiple help screens in your program, you will need to reset the help screen labels appropriately.

If you elect to set a branch to a help screen, this will temporarily override a TRAP command for help screen.

You will also be asked for a label to branch to on an "up arrow." This feature will allow you to control where program flow should go if the operator presses an **[up arrow]** key. If you do not enter a label here, program flow will branch to the previous "enter" command. You can also set a separate TRAP for up or down arrow. (See TRAP command.) Now the EDITOR will ask you if you wish to save this command. By responding "yes", the new command line will be inserted into your program.

Record number fields (see "R" type field in the Data Dictionary discussion) provide an interesting special case. If you use an ENTER command to reference an "R" type field, your program will automatically do a search for the specified record immediately after the user presses the **[RETURN]** key. If a record number field is left blank, the system will do a search for a 0 (zero) record number which has the effect of clearing the screen and memory buffers for this particular file.

Once created, the command line in your source code file will look like one of the following lines:

Enter (FIELD) mask (MASK VALUES) password type on help gosub (LABEL)

Enter (FIELD) password type

Enter (FIELD) mask (MASK VALUES) on help gosub LABEL

Enter (FIELD) mask (MASK VALUES) on up arrow goto LABEL

Only those subsets of the command that you selected will be displayed.

To write an ENTER command:

REQUIRED entry:	FIELD NAME
OPTIONAL entries:	LABEL MASK PASSWORD TYPE GOSUB HELP SCREEN ON UP ARROW GOTO

```
+----------------------+
|          =           |
|                      |
+----------------------+
```

PURPOSE:

The "equals" option allows you to define a field, value, or calculation and store it in a specified field. Once written, a new command line using "equals" might look like this:

FIELD 1=(FIELD2)+(FIELD3)

When this command is executed, FIELD1 will become equal to FIELD2 plus FIELD3.

USAGE:

This command is activated by pressing [=] or pressing **[RETURN]** when the field is highlighted. The EDITOR will display a prompt of 15 asterisks. Enter the name of the field into which the new calculation results are to be moved. Next the following subset of options will appear on your terminal:

Expression **L**ocation **P**ortion **T**rim **F**ill **A**ctual length **M**ax length **J**ustify
total **R**ecords **N**et (active) records

These options will be explained in detail on the following pages.

=

Expression

USAGE:

If you wish to move data from one field to another or perform a math calculation, use this "Expression" subset.

You should note this one special case: If you move a value into an "R" type (record number) field, the system will perform an immediate search to find the record associated with the value. This particular feature is unique to record number fields.

If you wish to move the value stored in FIELD2 into FIELD1 as indicated by the expression **FIELD1=(FIELD2)**, perform the following steps:

1) Select the = (equals) option by pressing [=]

2) Enter FIELD1 (the name of the field into which the new value is to be moved)

3) Select the Expression option by pressing **[E]** (or highlight "Expression" and **[RETURN]**)

4) Fill in the Expression line -- (FIELD2)

This will appear in your program as:

FIELD1=(FIELD2)

If you want to perform a math calculation and then move the results of that calculation into FIELD1, press the **[E]** key. The EDITOR will then display a prompt line where you can enter the expression. Here is an example of how an Expression type command line might appear in your program:

FIELD1=(FIELD2)+<10>*(FIELD3)

The SENSIBLE SOLUTION performs math calculations by starting at the left side of the expression and then working to the right. SENSIBLE SOLUTION math expressions differ slightly from math expressions found in syntax languages such as BASIC or Pascal.

There are four operators used in the SENSIBLE SOLUTION Language:

+ addition - subtraction * multiplication / division

The convention used in the Expression line to define the type of data to be manipulated are as follows:

( )'s must be used around field names
[ ]'s must be used around alphanumeric values -- literal strings
< >'s must be used around numeric values

Adding together two alphanumeric values (strings) provides a concatenated result. For example, adding [SKI] + [WEAR] results in SKIWEAR.

-- ARRAYS --

The SENSIBLE SOLUTION Language supports the use of one-dimensional arrays in any command that has an expression line. Both the "If" and "Equal Expression" commands have this capability. Arrays are indicated by the inclusion of an ampersand (&) in the expression line.

The SENSIBLE SOLUTION utilizes a flexible means of accessing arrays. A value stored in a field is used as an array counter to indicate which element of an array of similarly-defined fields is to be accessed.

An array of fields is allocated by assigning as many field names as desired in alphabetic sequence within the same file, all with the same data type and length. The SENSIBLE SOLUTION organizes fields within a file in ASCII order by field name. Assigning the names in sequence insures that the elements of the array are contiguous. One important point:

> If the names of the fields do not fall in order, the elements of the array may not be contiguous, and SENSIBLE SOLUTION will not be able to access the data correctly.

A particular element of the array is accessed by specifying the first element of the array followed by an array counter value. The system begins at the first element of the array then steps down through the buffer, by the length in bytes of the first element, until it reaches the desired element.

This allows the programmer to regard a subsection of an array as a valid array in itself simply by specifying the "first" array element as a field in the middle of the array. It is the programmer's responsibility to see that arrays are accessed in a valid manner. The system will happily hand you the

13th element of a 12-element array -- which is probably a meaningless chunk of the following field.

The array counter must be a variable stored in a field, not a numeric constant. In the expression line, the counter is always separated from the first array element field name by an ampersand (**&**).

If the array counter precedes the ampersand, and the field name follows the ampersand, then this will indicate that the value of the array element is to be read. For example, in the following expression, MONTHLYSALES01 is the first element of a 12-element array of sales-by-month. The total value of

(TOTALSALES)+(MONTH)&(MONTHLYSALES01)

is equal to "TOTALSALES" plus "MONTHLYSALES01". The value retrieved from the MONTHLYSALESnn array will correspond to the value stored in the array position specified by "MONTH".

If the ampersand is the first character in the expression line, and the array counter follows it, this would indicate that the array element is to be assigned a value. For example, incrementing MONTH from 1 to 12 and repeatedly executing the command

MONTHLYSALES01 = &(MONTH)<0>

would clear the MONTHLYSALESnn table to zero.

Arrays may also be both read and written to in the same expression:

MONTHLY SALES01 = &(MONTH)(SERVICECHARGE)+(MONTH)&(MONTHLYSALES01)

adds a constant service charge to the sales-by-month array value pointed to by the counter, "MONTH".

If the ampersand is to the left of the counter, it indicates that the FIELD NAME to the left of the = (equals) sign is the array. If the ampersand is to the right of the counter, it indicates that the next FIELD NAME in the expression line is an array. For example:

FIELD1 = &(COUNTER)<0> ampersand left of the counter --
array is left of the equal sign (FIELD1)

FIELD1 = (COUNTER)&(FIELD2) ampersand right of the counter --
array is right of the equal sign (FIELD2)

To write an EQUALS EXPRESSION command:

REQUIRED entries:	FIELD NAME EXPRESSION
OPTIONAL entry:	LABEL

```
+----------------------+
|          =           |
|       Location       |
+----------------------+
```

USAGE:

Use this command to determine the location of a particular character or value within a field. The value can be a constant (literal) or a variable. The location within the field will be passed back to a specified destination field. If the value or character is not found in the field being examined, a zero (0) will be returned to the destination field.

With this option you may do an in-string search for a single character or partial value. For instance, you could search a field for a comma and pass the location information back to the destination field. You could also search for the value "1234" within the string and pass back the beginning location within the field.

EXAMPLE:

Let's create the following command line to search a field for a comma and report its location:

FIELDNAME1 = the location of , within FIELDNAME2 starting at chr 1

Here is the layout of the screen display you will see as you create the new line in your program:

```
FIELDNAME1                    =location of  ,
within FIELDNAME2                        start at chr 001
```

1) Select the "equals" option by pressing [=].

2) Name your destination field **[FIELDNAME1]** then press **[RETURN]**.

Usually you will want to specify a temporary memory storage field. Instead of storing the value on the disk, temporary memory fields are used to retain information within a program; usually the value is changed during program execution. These temporary memory fields must be defined within the Data Dictionary and usually coincide with a "dummy" data file named MEMORY.MS/.KS.

3) Press **[L]** for the command subset "Location" and the screen will display

Location of ***************

Constant **Field**

4) Specify the type of value for which you are searching. Press [C] to search for a constant value or press **[F]** to search for a value stored in another field. Since we are searching for a comma, we will specify "Constant."

5) Enter the value of the constant to be found -- in this case it would be **[,]** and then press **[RETURN]**. If you had selected "F" (Field) on the prior option, you would enter the name of the field containing the value for which you are searching.

6) Now the prompt line will display:

within ***************

Enter the field to be searched -- **[FIELDNAME2]** and press **[RETURN]**.

7) Next the EDITOR will request:

start at chr???

Constant **Field**

Specify that this is a constant value by pressing **[C]**.

8) Enter the position within the field at which you want the search to begin. Since we want to search the entire string, we will enter a **[1]** and press **[RETURN]**. If you had selected **[F]** (Field) you would enter the FIELD NAME storing the character position number at which you want to begin your search.

9) Enter **[Y]** or press **[RETURN]** to save the line you have just written.

The completed program line will look like this:

FIELDNAME1 = the location of , within FIELDNAME2 starting at chr 1

To write an EQUALS LOCATION command:

REQUIRED entries:	DESTINATION FIELD NAME
	VALUE TO BE SEARCHED FOR -- CONSTANT OR FIELD
	FIELD TO BE SEARCHED -- FIELD NAME
	CHARACTER NUMBER POSITION TO BEGIN SEARCH
OPTIONAL entry:	LABEL

```
+----------------------+
|          =           |
|       Portion        |
+----------------------+
```

USAGE:

This command is used to parse a field and extract a portion of it. The portion extracted will be loaded into the specified destination field. The command retreives a substring of a field and moves that substring to the destination field.

EXAMPLE:

FIELDNAME1 = portion of FIELDNAME2 from FIELDNAME3 for FIELDNAME4 chars

Here is the layout of the screen display as you create this new line in your program:

```
FIELDNAME1                  = portion of FIELDNAME2
start at pos FIELDNAME3                  for num of chars FIELDNAME4
```

1) Select the "equals" option by pressing [=].

2) Name the destination field -- **[FIELD1]**.

3) Select the subset option **[P]** (Portion) and you will then see

   portion of ***************

   on your screen.

4) Enter the name of the field from which to extract the substring **[FIELDNAME2]**.

5) You will now need to specify the character position from which the search will begin. This can be a constant or a field. If a field

name is chosen the value in the field can be altered under program control. The EDITOR will now display

start at pos****************

Constant Field

and requests that you choose whether this is a field value or a constant. If you specify that this location is a field, you must define the field in your program. Make your selection; in this case, press **[F]**.

6) Now name the field which is the beginning position of the substring which you want to retrieve. **[FIELDNAME3]**. Note that the starting character number is the byte position of the beginning of the substring. The first character of any field is position **1**. If you had chosen **[C]** (Constant) name the character position within the field at which you will begin your extraction.

7) Next, specify how many characters to extract. Once again this could be a constant or a field. If a field name is chosen, the value in the field can be altered under program control. In our example we will use a field so the EDITOR will now display:

****

where you must enter the name of the field containing the value equal to the number of characters to be extracted.

8) If you designate the length of your desired substring as a constant, enter the number of characters in the string and press **[RETURN]**.

9) Press **[Y]** or **[RETURN]** to save your new command line.

When using the Portion command, all data types may be considered as ASCII strings. The "day" portion of a date, for example, is a substring beginning at position **4** (mm/**dd**/yy) which is 2 characters long.

To write an EQUALS PORTION command:

REQUIRED entries:	DESTINATION FIELD NAME
	FIELD TO BE PARSED
	POSITION TO BEGIN EXTRACTING -- CONSTANT OR FIELD
	NUMBER OF CHARACTERS TO EXTRACT -- CONSTANT OR FIELD
OPTIONAL entry:	LABEL

=
Trim

USAGE:

This command allows you to trim the leading or trailing blanks from a field.

Note! You may want to move the field being trimmed to a temporary memory field. Trim the spaces from the temporary memory field. This will leave your original field unchanged by the "trim" command. If you do not move the field to a memory field, trimming blanks from the field will cause any record saved back to the data file to be offset improperly. This would result in garbled field values when the record is viewed on screen.

EXAMPLE:

```
FIELDNAME = CUS.NAME
FIELDNAME = trim spaces trailing
```

Suppose you are interested in sending a form letter to each of your customers addressing them by their first name. Obviously you will need space in your CUS.NAME field to hold a name like Montague (8 characters), but if your customer's name is Chuck (5 characters) you'll need a way to trim the last three spaces provided in your name field or you'll have a salutation line that looks like this:

```
                              Dear Chuck   ,
instead of like this:
                              Dear Chuck,
```

There is a command to solve this problem; it works by allowing you to trim "Leading" or "Trailing" blanks.

To write an EQUALS TRIM command:

REQUIRED entry: FIELD NAME
SPECIFY LEADING OR TRAILING

OPTIONAL entry: LABEL

```
+----------------------+
|          =           |
|         Fill         |
+----------------------+
```

USAGE:

Use this command to fill the leading or trailing blanks in a field with a specified character.

EXAMPLE:

FIELDNAME = fill leading chrs with

This command will allow you to specify a fill character for leading or trailing blanks in a field.

Suppose you are printing checks and do not want to leave blanks in front of the dollar amount. You could use the "Fill" option to place any character that you wish as leading characters in your dollar amount field. If you specified an asterisk as the character, the value may be printed out like this:

```
*******575.55
```

To write an EQUALS FILL command:

REQUIRED entries:	FIELD NAME SPECIFY LEADING OR TRAILING CHARACTER TO FILL WITH
OPTIONAL entry:	LABEL

=
Actual length

USAGE:

Use this command to check the number of significant characters in a field and to send that information to a specified destination field. You should note, however, that spaces are considered to be significant characters. Consequently, you may need to trim trailing or leading spaces prior to writing an "Actual length" command line.

Note! You may want to move the field being checked to a temporary memory field. Trim the spaces from the temporary memory field and then check the "actual length." This will leave your original field unchanged by the "trim" command. If you do not move the field to a memory field, trimming blanks from the field will cause any record saved back to the data file to be offset improperly. This would result in garbled field values when the record is viewed on screen.

EXAMPLE:

```
FIELDNAME = trim spaces trailing
FIELDNAME = actual length of CUS.NAME
```

If you have a name field with a defined length of 30 characters and you have a name in it that is 17 characters long, the integer 17 will be passed to the destination field.

To write an EQUALS ACTUAL LENGTH command:

REQUIRED entries:	MUST FOLLOW AN EQUALS TRIM COMMAND DESTINATION FIELD NAME FIELD TO BE CHECKED
OPTIONAL entry:	LABEL

=
Max length

USAGE:

This command will pass a value to a destination field equal to the maximum length of a specified field.

EXAMPLE:

FIELDNAME = maximum length of field CUS.NAME

If you have a CUS.NAME field that will hold thirty characters, this command will pass back the number 30 to your specified destination field.

To write an EQUALS MAX LENGTH command:

REQUIRED entries:	DESTINATION FIELD NAME FIELD TO BE CHECKED
OPTIONAL entry:	LABEL

=
Justify

USAGE:

Use this command to justify the significant characters within the field. You can left justify, right justify, or center the significant characters.

EXAMPLE:

CUS.NAME= justify left

A name field of 30 characters, with the name SMITH, WILLIAM in it, would appear as follows with the different justifications. (*'s equal blanks in this example.)

left justified	SMITH, WILLIAM****************
right justified	****************SMITH, WILLIAM
centered	********SMITH, WILLIAM********

Within a record, a string field is normally left justified and a numeric field right justified.

To write an EQUALS JUSTIFY command:

REQUIRED entries: FIELD NAME
                     SPECIFY JUSTIFY -- LEFT, RIGHT, OR CENTER

OPTIONAL entry: LABEL

```
+----------------------+
|     =                |
|     total Records    |
+----------------------+
```

USAGE:

Use this command to assign a value equal to the total number of records in a specified file to a destination field. The number of records returned will include any deleted records in the file.

EXAMPLE:

> **FIELDNAME = total number of records in FILENAME**

To write an EQUALS TOTAL RECORDS command:

REQUIRED entries:	DESTINATION FIELD NAME FILE NAME
OPTIONAL entry:	LABEL

=
Net (active) records

USAGE:

With SENSIBLE SOLUTION data files, records exist in one of two different states -- active or deleted. If a record has been deleted from a file it is not physically erased; it is simply flagged as deleted or inactive. Deleted records are then reused by SENSIBLE SOLUTION as they are needed.

The NET ACTIVE RECORDS command is used to assign a value, equal to the total number of active records in file, to a specified destination field. The number of records returned will include only active records; deleted records will be excluded.

EXAMPLE:

```
        FIELDNAME = tot num of active records in FILENAME
```

To write an EQUALS NET ACTIVE RECORDS command:

REQUIRED entries:	DESTINATION FIELD NAME FILE NAME
OPTIONAL entry:	LABEL

If

PURPOSE:

Use this command to make a logical comparison of two fields. You may use arrays with this command.

USAGE:

When you select the **If** command, the conditions for which you can test appear in a menu like the one below:

```
   1  <      2  <=     3  =     4  =>     5  >     6  <>
   Duplicate key check    Record not active check
```

The first 6 options are symbols representing the following:

1) less than
2) less than or equal to
3) equal to
4) equal to or greater than
5) greater than
6) less than or greater than

After you select one of the options from 1 through 6 you will be prompted to enter the name of the field being tested:

```
                  if ****************
```

After entering the field name, an expression line will be provided for you to enter either the field name you wish to compare or a literal value.

```
   if FIELD1             (test condition)

*******************************************************************
```

Regarding the expression line, the same conventions apply here as in the = (equal) command:

( )'s around field names
[ ]'s around alphanumeric literals
< >'s around numeric literals

You can compare a field to a math calculation in the expression line and you can also use arrays with this command.

Once you have entered the expression line, you are asked if the branch is a GOTO or GOSUB. Make your selection and then enter the line label to which program flow will divert. You will see the following prompts on the screen:

```
          if FIELD1         (condition specified  go(to)(sub)  LABEL
          FIELD2
goTo  goSub
```

A sample command line might be:

```
if FIELD1 < (FIELD2) gosub LABEL
```

```
+------------------------+
|  If                    |
| (test condition) goTo  |
+------------------------+
```

-- If Less Than Goto
-- If Less Than or Equal Goto
-- If Equal Goto
-- If Greater Than or Equal Goto
-- If Greater Than Goto
-- If Not Equal Goto

PURPOSE:

These commands compare the value of a field to the value of an expression. The test is: FIELD (test) EXPRESSION. If the test is true, program control is transferred unconditionally to the specified line label. Otherwise, program control "falls through" to the command following the test.

USAGE:

Here is a layout of the screen display you will see as you create the command:

```
if FIELDNAME1      (condition specified)     goto LABEL
  FIELDNAME2
```

Enter the name of field which is to be compared with the value of the calc expression.

Enter the argument for the expression line. This can be a literal, field, calculation, or an array element.

Enter the GOTO "target" line label where the program will branch to if the test is true.

To write an IF (TEST) GOTO command:

REQUIRED entries:	TEST CONDITION FIELD TO BE TESTED -- FIELD NAME EXPRESSION TO TEST AGAINST BRANCH OPTION -- GOTO COMMAND LABEL BRANCHED TO
OPTIONAL entry:	LABEL

If
(test condition) goSub

-- If Less Than Gosub
-- If Less Than or Equal Gosub
-- If Equal Gosub
-- If Greater Than or Equal Gosub
-- If Greater Than Gosub
-- If Not Equal Gosub

PURPOSE:

These commands compare the value of a field to the value of an expression. If the value of the field and the expression match the specified line label is called as a subroutine. Otherwise, program control "falls through" to the command following the test.

USAGE:

Here is a layout of the screen display you will see as you create the command line:

```
if FIELDNAME1      (condition specified)     gosub LABEL
   FIELDNAME2
```

Enter the name of the field which is to be compared with the value of the calc expression. Next, enter the argument for the expression line. This can be a literal, field, calculation, or an array element. Finally, enter the GOSUB "target" line label (where the program will branch to if the test is true.)

A "RETURN" command line must be placed at the end of a subroutine.

To write an IF (TEST) GOSUB command:

```
REQUIRED entries:    TEST CONDITION
                     FIELD TO BE TESTED -- FIELD NAME
                     EXPRESSION TO TEST AGAINST
                     BRANCH OPTION -- GOSUB
                     COMMAND LABEL BRANCHED TO

OPTIONAL entry:      LABEL
```

EXAMPLES:

```
if FIELD1 = (FIELD2) gosub LABEL
                        or
                     goto LABEL

          or

if FIELD1 = (FIELD2)+<10> gosub LABEL
                             or
                          goto LABEL

          or

if FIELD1 = (counter)&(FIELD2) gosub LABEL
                                  or
                               goto LABEL
```

where FIELD2 is an array and the counter is set to a value representing the element of the array you want to compare. FIELD1 can also be an array and the command would look like:

```
if FIELD1 = &(counter1)(counter2)&(FIELD2) gosub LABEL
                                              or
                                           goto LABEL
```

This command would compare the element of the (FIELD1) array, as defined by (counter1), to the element of (FIELD2) array as defined by (counter2).

```
if FIELD1 = &(counter1)(counter2)&(FIELD2)+<5>
```

This command line would compare an element of the (FIELD2) array, plus 5, with the element of the (FIELD1) array.

```
+----------------------+
| If                   |
| Duplicate key check  |
+----------------------+
```

PURPOSE:

This command will check to determine if a key field value in an existing data file matches a value entered by the user. If a field value already existing in the file matches the value entered by the user, then program control will be transferred to a specified line label. Otherwise, control "falls through" to the command following the duplicate key check.

USAGE:

Here is a layout of the screen display you will see as you write the command:

if duplicate key exists for KEY FIELDNAME   goto LABEL

First, name the key against which the test is to be made.

Enter the Goto label. This is the "target" line label to which control will be passed if the value already exists in the key field.

If a duplicate key is found, the message

A record already exists with value entered

will be displayed at the bottom of the operator's screen. If the operator presses **[RETURN]** program flow will branch to the specified label.

This command is usually used to prevent multiple entry of keys into a file. The operator will receive an error message if the field is not a key field and therefore cannot be checked.

A duplicate key check will never trigger a file trap.

To write an IF DUPLICATE KEY EXISTS command:

REQUIRED entries:	KEY FIELD NAME COMMAND LABEL BRANCHED TO
OPTIONAL entry:	LABEL

If
Record not active check

PURPOSE:

With SENSIBLE SOLUTION data files, records exist in one of two different states -- active or deleted. If a record has been deleted from a file it is not physically erased; it is simply flagged as deleted or inactive. Deleted records are then reused by SENSIBLE SOLUTION as they are needed.

This command tests the "current record" to see if it is still active. If the current record number has been cleared (is zero), or a "find record" is unsuccessful, program control will pass to a specified line label. Otherwise, control "falls through" to the command following the test.

USAGE:

Here is a layout of the screen display you will see as you create the command line in your program:

```
if record not active file FILENAME    goto LABEL
```

Name the file to be tested for an active current record.

Enter the Goto label. This is the "target" line label to which control is passed if the "current record" is not active.

To write an IF RECORD NOT ACTIVE GOTO command:

REQUIRED entries:	FILE NAME COMMAND LABEL TO BRANCH TO
OPTIONAL entry:	LABEL

```
+---------------------+
|         Go          |
|                     |
+---------------------+
```

PURPOSE:

GO is used to divert program control to a new label in the program.

USAGE:

The **GO** command group is activated by pressing [G] or [**RETURN**] when the field is highlighted.

The **GO** command group is comprised of four options:

go**T**o LABEL

go**S**ub LABEL

Goto save group

Return

Go
goTo

PURPOSE:

This command will allow you to branch program control to a specified label.

USAGE:

When you select goTo by pressing **[T]** or highlighting the option and pressing **[RETURN]** you will see

```
goto            **********
```

Enter the command line label to which you want program control to divert. Now you will be asked:

```
go to line depending on the value of the field?  yes  no
```

When you answer "yes" to this question you can create a command line that transfers control to one of several different program lines depending on the value of a field. If the field value is less than 1, or exceeds a specified maximum, control will be passed to a specified "goto" line label.

If you answer "yes" the EDITOR will ask:

```
on the value of ***************
```

Enter the name of the field whose value will be tested. If the field value is 1, control will be transferred to the command immediately following the **goTo.** If the value is 2, the second command will be executed, and so on. Be sure that you have a value in the field before performing the goto command.

Next, enter the maximum number of goto's that you wish to use in this command line. If the value of the field exceeds this maximum value (or if it is less than 1), control will be passed to the line label specified as your target label.

This command is generally used in menu programs. However, it may be used within a program to do conditional branching based on the value of a specified field. The function of "conditional goTo" is to allow the programmer to selectively branch to another program line.

EXAMPLES:

A sample menu program --

```
START enter N.1.0.1
      goto line on value of N.1.0.1 maximum gotos 03 if error goto START
      execute .RUN FILE 1
      execute .RUN FILE 2
      execute .RUN FILE 3
```

A non-menu program --

```
START enter N.1.0.1 mask [1 3 5]         (accepts only 1,3, or 5 values)
      goto line on value of N.1.0.1 maximum gotos 05 if error goto START
      FIELD = <VALUE 1>
      goto NEXT STEP
      FIELD = <VALUE 2>
      goto NEXT STEP
      FIELD = <VALUE 3>
NEXT STEP  (continue with the next command line in your program)
```

To write a GOTO command:

REQUIRED entry: LABEL OF COMMAND LINE BRANCHED TO

OPTIONAL entries: TRANSFER VALUE FIELD
MAXIMUM GOTOS
LABEL

```
+-----------------------+
|        Go             |
|        goSub          |
+-----------------------+
```

PURPOSE:

This command transfers program control to a specified line label but "marks its place" so that a subroutine "Return" command can transfer control back to the next command. By using a gosub command, you can temporarily branch to a subroutine in your program and then return to your original command path.

USAGE:

When you select the option goSub by pressing **[S]** or highlighting it and pressing **[RETURN]** the screen will show:

```
gosub         **********
```

The EDITOR is asking for the label of the "target" line. Each time you insert a command line into your program, the first thing the EDITOR does is offer you the opportunity to label that line. This line label acts as a target. When you write a gosub command, the EDITOR asks you to name the target line label for program control to branch to. Enter this label and press **[RETURN]**.

To return from a gosub (subroutine) you must use the "Return" command, not a goto command. You will be returned to the next sequential command following the gosub.

When you execute a gosub command, the location of the next sequential command in your program is loaded into a stack register. When the Return command is executed, the address at the top of the stack is used to point to the command to return to.

The stack register can hold up to 20 pointers. This allows for nested subroutines 20 deep.

To write a GOSUB command:

REQUIRED entry: COMMAND LINE LABEL TO BRANCH TO

OPTIONAL entry: LABEL

```
+---------------------+
|   Go                |
|   Goto save group   |
+---------------------+
```

PURPOSE:

This command is commonly used to trigger a "save record" action for a group of data files under program control, without requiring the user to press **[ESC] [S]** (save record). This command can also be used to request a response from the operator before performing calculations and updates to records prior to saving. Suppose the operator has input all of the information necessary for the program to make calculations and update the files. If the operator has made an entry error, he has the opportunity to not save the information but, instead, to correct the error. All of the calculations are done after the operator confirms the save, not while the operator is entering the information.

USAGE:

Using this command will cause the message "SAVE this record? (Y/N)" to be displayed on the operator's screen; only a single keystroke will be accepted for an answer. The keystroke is automatically converted to upper-case. If the answer is **Y**, control is transferred to the line label specified by the command; otherwise, control is passed to the command following the last executed "mount screen" command. This "SAVE RECORD?" question will bypass any SAVE WITH CONFIRM command lines immediately following this command line.

When you write the command "Goto save group" by pressing **[G]** or highlighting it and pressing **[RETURN]**, the screen will show:

goto save group **********

Enter the line label to which control will be transferred if the answer to "SAVE this record? (Y/N)" is **Y**.

SAVE WITH CONFIRM commands normally ask the "SAVE?" question for each command. If you have a group of saves involving several different files, the multiple requests can become tedious. The "Goto save group" command can eliminate this problem by asking the "SAVE?" question once; subsequent requests for confirmation will be suppressed until a command other than a "save" is encountered.

To write a GOTO SAVE GROUP command:

REQUIRED entry: COMMAND LINE LABEL TO BRANCH TO

OPTIONAL entry: LABEL

Go
Return

PURPOSE:

This command is used to mark the end of a subroutine and to transfer program control back to the command immediately following the Gosub that called the subroutine.

USAGE:

A Return command must end every Gosub routine. An error message will be displayed if you encounter a Return without first having executed a Gosub.

To write a RETURN command:

REQUIRED entry:	NONE
OPTIONAL entry:	LABEL

```
+---------------------------+
|          Mount            |
|                           |
+---------------------------+
```

PURPOSE:

This command will display a SENSIBLE SOLUTION screen file (filename.SCC). Use the "Mount" command to display both report formats and screen formats. The number of screens that can be mounted is only limited by the amount of available memory.

USAGE:

When you select the option **M**ount by pressing **[M]** or highlighting it and pressing **[RETURN]**, the screen will show:

Screen format   Reporter format

If you select "Screen format" by pressing **[S]**, the EDITOR will provide a field in which you must enter the name of the screen to be displayed. Multiple screens may be used in a single program and the screens may be standard "Screen formats" or "Reporter formats". This allows you to print multiple detail lines on the screen or interactively input data through one or more screens and then print out reports -- all with one program.

If you choose to mount a "Reporter format" screen by pressing **[R]**, you must enter the name of the Reporter format. The EDITOR will then ask you to identify where the print out is to be sent. Select one of these four options:

Ask at run time Printer Crt Disk

If you select "Ask at run time", a message will be displayed on the operators screen requesting them to choose where the report output should be sent -- either to the Printer, the CRT, or the Disk.

If you select "Disk", you will be asked to name the file to which you want the program to send the output. Carriage returns will be automatically added to the end of each format line. You can use this option to send data to the disk that can then be used as input for word processing or spread sheets. See the PRINT command.

EXAMPLE:

mount report format FILENAME print on ask at run time

If you are sending control characters to the printer or terminal, you must be certain to have the output device defined. If no output device is assigned (Ask at run time), then the operator must determine where the control characters will be sent by his selection.

If a Reporter format screen is not MOUNTED, the system assumes that all control characters will be sent to the terminal. Direction of output can be established and re-established by mounting and re-mounting format screens. The same format screen can be remounted in a single program.

To write a MOUNT command:

```
REQUIRED entry:  SCREEN OR REPORTER FORMAT NAME

                 for REPORTER format only --
                 DIRECTION OF PRINT OUT -- ASK AT RUN TIME, PRINTER, CRT,
                                           OR DISK

OPTIONAL entry:  FILENAME for "print to disk" option
                 LABEL
```

Save rec

PURPOSE:

Use this command to save a the record in a specified file.

USAGE:

After selecting the "Save record" option by pressing [S] or highlighting it and pressing [**RETURN**], you will be asked to name the file to which you want the record saved. Next, the EDITOR will display on the screen:

Confirm YES NO

You have the option of forcing the user to confirm that they want to save the record. If you choose "yes", the user will see this message on their screen:

SAVE this record? (Y/N)

The default answer is "yes." If the user should answer the question "no," the program will pass control to the first "Enter" command following the "Mount screen" command (the first input field on the screen).

On the other hand, if you specify that you do not want confirmation from the user (no confirm), the record will be saved without a request. This provides "automatic" data update without operator intervention.

Now the EDITOR will ask you to specify:

Clear buffer YES NO

If you answer YES, both the screen and the memory buffer will be cleared after each "save" command is executed. After a record is saved, all fields of this record are cleared to zero or blanks in internal buffers, displayed fields are cleared, and the record number will be set to zero.

If you select the "confirm" option, you can also enter the line label to which you want program flow to branch if the user does not wish to save the record. On screen you will see:

if no save then goto **********

Specifying a goto label will allow you to control program flow. Control will pass to the labeled command line, instead of going to the command line following the "Mount screen" command. When you execute a search for a record and update that record, saving it back in the file will overwrite the old record. If there was no record in the buffer (you are creating a new one), by saving it, it will append this new record to the data file.

Choosing NO means that buffers and screen display are not cleared of data, and the current record number remains unchanged. The option to not clear buffers would be used when you not only wish to save data in a file, but also want the data available for further processing, such as a **(FIND next record) [ESC] [N].**

The system determines what is happening at the time a "save rec" command is executed by checking the number of the record. If a value greater than zero (0) exists in the record number field, the system recognizes that you are updating an existing record and writes it in that position. Otherwise, it recognizes the record as a new one and either overwrites the first deleted record in the file or appends the new record onto the end of the file.

EXAMPLES:

```
save rec in file (FILENAME)  confirm/clear buffer
save rec in file (FILENAME)  no confirm/no clear buffer
```

To write a SAVE RECORD command:

REQUIRED entry: DESTINATION FILE NAME

OPTIONAL entries: CONFIRM OR NO CONFIRM
CLEAR BUFFER
COMMAND LINE LABEL TO GOTO IF NO SAVE
LABEL

Delete rec

PURPOSE:

This command is used to delete a record from a specified file.

USAGE:

delete rec in file FILENAME confirm if no delete then goto LABEL

After selecting the DELETE RECORD option enter the name of the file from which you want to delete a record. The EDITOR will ask:

Confirm   No confirm   Delete all records

The sub-option "Confirm" will generate a message to the operator asking "REMOVE this record? (Y/N)". The default answer is NO. If the user answers NO, control is transferred to the first "Entry" command following the "Mount screen"command. If the user answers YES, the current record will be removed from the data file. In addition, the record buffer and terminal display will be cleared. The "current record number" for this data file will be reset to zero.

If you select the "confirm" option, you can enter the command line label to which you want program flow to branch if the user responds NO to a "REMOVE this record?" message.

Since the "current record number" is reset to zero, operator entries following a delete will be considered new entries. If the operator wants to modify existing records, they will be required to perform a "Find" first. The operator will get an error message if they attempt to delete a record when the "current record number" is zero.

delete rec in file FILENAME no confirm

Specifying the sub-option "No confirm" will delete the record without asking the operator; the buffers and the operator's screen display will be cleared, and the "current record number" will be reset to zero.

Note that "automatic" data deletion is provided by this command. This command is **very** dangerous, for obvious reasons, and should be avoided in programs where it is not absolutely necessary.

delete rec in file FILENAME all recs

The sub-option "Delete all records" will initialize the file (see SENSINIT.COM). Thus, it will appear that all records have been deleted. Actually, the command sets the "number of records", a value maintained in the .KS file, to zero. When this is completed, program control will return to the next line in the program following the "delete rec" command line. This command will usually be used in utility programs such as month's-end cleanup and the like.

To write a DELETE RECORDS command:

REQUIRED entry: FILENAME CONTAINING RECORD TO DELETE

OPTIONAL entries: CONFIRM
NO CONFIRM
ALL RECORDS
COMMAND LINE LABEL TO BRANCH TO
LABEL

```
+---------------------+
|        Clear        |
|                     |
+---------------------+
```

PURPOSE:

This command is used to either clear the values displayed on the operator's screen (clear the buffer) or to clear the current record number from memory.

USAGE:

When you select the Clear option by pressing **[C]** or highlighting the option and pressing **[RETURN]**, the following options will appear on your screen:

```
            clear

     Buffer    Record
```

These options will be explained in detail on the following pages.

Clear Buffer

PURPOSE:

"Clear buffer" is used to clear the screen and memory buffer of records that are currently active. This command clears the record buffer for the specified data file, sets the "current record number" to zero, and empties the field windows on the terminal display.

USAGE:

clear buffer in file FILENAME

After selecting this option, enter the name of the data file you wish to clear. Data on the screen and in the file buffer will be cleared. This command can be used to insure that data will not be left on-display or in buffers, against the possibility of an accidental "save" or "delete". It is also used to clear specific files out of a screen where there are multiple files being displayed.

The "clear fields" option (**[ESC] [C]**) executed by the operator will clear all file buffers used in the program unless a trap is set to clear only specified files.

A TRAP can be set for an operator initiating a "clear screen" from the keyboard. If a trap is set, you can branch to a subroutine in the program where you have control over which buffers are cleared. This would be used where multiple files are being displayed in a screen and you want only certain ones cleared. (e.g., You have a header record on the screen and are using values from it to find related transaction records. You might want to clear the transaction records but not the header record.)

For more information see the section on TRAP command for "clear".

Clear
Record number

PURPOSE:

This command will clear the current record number from memory. The data in the record will remain in memory.

USAGE:

clear record number in file FILENAME

After selecting this option, enter the name of the data file corresponding to the record number you wish to clear.

If you are reading records with locks, this command has the function of unlocking the record. This allows you to retain the information in the record yet make it available to other users in the system.

Usually this applies to a shared system file that many application programs use to obtain information. Fields like date, company name, invoice numbers, etc., would typically be used by a number of application programs in an accounting system; this information is usually kept in a one record file called a "system" file.

Since this record must be available to each application, you can read the record in locked and then clear the record number field -- thus making it available to other users.

If you are updating information in this record, your program will will need to reread the record with a lock into memory, update it, save it back to the disk, and then clear the record number to unlock it again.

A second function performed by clearing the record number affects saving the record. When the system saves a record, it checks the record number field to see if a value exists. If one does, the save will rewrite the record to the disk on top of the old record with the same number. If no value exists in the record number field, the save will append the record to the end of the data file.

You should be cautious about clearing the record number field when you are trying to update and save the same record.

To write a CLEAR RECORD NUMBER IN FILE command:

REQUIRED entry:	FILE NAME
OPTIONAL entry:	LABEL

Find

PURPOSE:

The "Find" command allows you to specify different techniques by which records will be retrieved from data files. With the options available, you have very flexible, powerful, and fast search capabilities.

OPERATION:

All "Finds" (whether executed as commands from within a .RUN file, or triggered from the flash menu during entry to a field) operate in the same way:

1) The FILETRAP target is marked for possible use. This may be a **local** filetrap (specified in the FIND command), the current **global** filetrap, or **none** if trapping is currently disabled.

2) If the find is a NEXT or PREVIOUS, the "last key accessed" is looked up. If there is no "last key accessed" SENSIBLE warns "Must SEARCH before doing NEXT/PREV" and either execution continues with the next command in the .RUN file (with FIND commands), or the flash menu is redisplayed (with ENTER commands).

3) If the find is on a **key field** (not a record number), the FILE INFORMATION SECTOR (physical record #0 of the .KS file) is locked in a multi-user system. This prevents anyone else from doing any finds, saves or deletes while you are looking up your key. If another user has the FILE INFORMATION SECTOR locked already, your program waits until it can achieve a lock. You cannot escape from this wait loop.

4) The "find" is performed.

5) The FILE INFORMATION SECTOR is unlocked. Typically, the entire lock/find/unlock cycle takes less than a second on a hard-disk system, so you will seldom contend for use of the FILE INFORMATION SECTOR.

System Operation Following a Successful Find

When a find is succesful, the found key is marked as the "last key accessed" for this file, and this file is flagged as the "last key file accessed".

Subsequent NEXT/PREV searches from the keyboard (i.e. from an ENTRY flash menu) will use this key for the search criteria.

Then the looked-up record is read from the .MS file. Now the system

1. Attempts to lock this record. If another user has this record locked already, a message is displayed. The user may ESCAPE to "top-of-screen" (first command after the last-executed MOUNT command), or wait until the existing lock is released. If the user waits, the message will clear as soon as his lock is gained.

   This lock will be held until 1) the record number is cleared, 2) the record is deleted, 3) an unsuccessful find is attempted, or 4) the user exits the .RUN file.

2. Inspects the looked-up record to see if it is flagged as "deleted". This should not happen with a keyfield lookup but might happen if the lookup is done by record number. A "deleted" record clears all fields to spaces and marks "current record number" as zero.

3. Displays on the current screen the contents of the fields in the record that was found.

If the RELATED trap is set and a successful "find" occurs, the system will perform a specified GOSUB and then return to the next line in the progam.

Execution continues with the next command in the .RUN file (with FIND commands). When the "find" was initiated through keyboard entry from the flash menu, the flash menu is redisplayed (with ENTER commands).

System Operation Following an Unsuccessful Find

When a find is unsuccessful, the system

1. Clears the "last key accessed" (if the find was not a NEXT or PREVIOUS).

2. If a TRAP "F" (file error) is set, the system continues program execution from the specified line label, leaving any current record untouched (i.e. a record looked up on an earlier find).

3. If the find failed by encountering a "beginning of keys" or "end of keys" condition, the current record will remain untouched; otherwise, the record CONTENTS are unchanged but the record NUMBER is cleared. This releases the lock on the record and makes subsequent saves "new" saves rather than "replace" saves.

Execution continues with the next command in the .RUN file (with FIND commands). When the "find" was initiated through keyboard entry from the flash menu, the flash menu is redisplayed (with ENTER commands).

USAGE:

This command can be activated by pressing **[F]** or pressing **[RETURN]** when the field is highlighted.

The following subset of options will appear on your terminal:

```
Exact   Generic   Related   First   Last   Next   Previous
```

These options will be explained in detail on the following pages.

Find Exact

PURPOSE:

This command will search for a record exactly matching the value in the field specified. The associated record will be read into the memory buffer and whatever fields are on the screen will be displayed.

USAGE:

find exact rec using field FIELDNAME on error goto LABEL

Name the field on which the search is to be done. A value must be present in the field before this command is executed by your program.

Then, specify a line label designating a subroutine to which you want program control diverted in the event of an unsuccessful search. An error message will be displayed if the specified field is not an index-key field. An error message will also be displayed if an exact match is not found and an "on error goto" label is not used in your program. The operator must press the [RETURN] key to continue processing.

The label specified in this command will override a TRAP "F" (File Error) command. If no label is specified, the TRAP would be activated. See TRAP "F" (File Error) in the section on TRAP commands.

To write a FIND EXACT RECORD command:

REQUIRED entries:	FIELDNAME ON WHICH TO SEARCH
OPTIONAL entries:	COMMAND LABEL TO BRANCH TO LABEL

Find Generic

PURPOSE:

This command is similar to "find record by exact field" except that if an **exact** match is not found, the search returns **the next higher sequential record** in the file. For example, a search on "SMI" might return "SMI", "SMITH", "SMOOCH" or "ZOOLOGICAL SOCIETY", depending on which was the "next-higher neighbor" in the index. The associated record will be read into the memory buffer and the appropriate field windows on the screen will be filled with data.

USAGE:

find generic rec using field FIELDNAME on error goto LABEL

First, name the field on which to search. Next, specify a line label designating a subroutine where you want program control to branch to in the event of an unsuccessful search. An error message will be displayed if the specified field is not an index-key field. If no record equal to or higher than the specified value is found, an END OF FILE ENCOUNTERED message will be displayed.

A TRAP "F" (File Error) will be activated only by an END OF FILE condition; program flow will branch to the goto label set in the TRAP command line. If an "on error goto" label has been specified, it will override the TRAP label. See TRAP "F" (File Error) in the section on TRAP commands.

To write a FIND GENERIC RECORD command:

REQUIRED entries:	FIELDNAME ON WHICH TO SEARCH
OPTIONAL entries:	COMMAND LABEL TO BRANCH TO LABEL

Find Related

PURPOSE:

This command will locate a "related" data record -- one for which the value of a specified field exactly matches that of the search field. The associated record will be read into the memory buffer and the appropriate field windows on the screen will be filled with data.

This command is most often used to locate data in files when the operator is entering data.

Note! A "RELATED RECORD" find works like an "EXACT" find (including trigger of any local or global filetrap), except for one thing:

A successful "related" find sets the "last key accessed" for the related file but **does not** change the "last key file accessed". This means that the **command** "FIND NEXT/PREV in related file" will work, but "FIND NEXT/PREV" **from the keyboard** will look up on the PRIMARY file, not on the related file.

USAGE:

find rec using field FIELDNAME related field FIELDNAME on error goto LABEL

Begin by entering the field name. The value of this field is the value to be searched for in the other file. It need not be a key but it **must** be a field not a constant. A value must be present in this field before the command line is executed.

Next, name the field on which the search is to be conducted. This field **must** be a key field and must be in the file being searched.

Finally, specify a line label designating a subroutine where you want program control to branch to in the event of an unsuccessful search. An error message will be displayed if the designated field is not a key-field. An error message will be displayed if a related record is not found. The operator must press the **[RETURN]** key to continue processing.

When you are accessing multiple related data files, you can set up the "R" TRAP to trigger a GOSUB to a subroutine which does related record searches whenever one of the "search" functions (Find, find Next, find Beginning etc.) is initiated by the operator.

Take heed! If you have set an "R" (Relates) TRAP, be careful not to have your File Error trap take you out of the RELATES subroutine and circumvent the "return" command of the RELATES subroutine. This will finally result in a "STACK FULL" error.

The TRAP "F" (File Error) will be activated if set and program flow will branch to the goto label set in the TRAP command line. If an "on error goto" label has been specified in the command, it will override the TRAP goto label.

See TRAP "F" (File Error) and "R" (Relates) in the section on TRAP commands.

To write a FIND RELATED RECORD command:

REQUIRED entries:	FIELDNAME CONTAINING VALUE TO FIND FIELDNAME TO BE SEARCHED
OPTIONAL entries:	COMMAND LABEL TO BRANCH TO LABEL

```
+----------------------+
|    Find              |
|    First   Last      |
+----------------------+
```

PURPOSE:

These two commands find the **lowest** and **highest** index key values for the field specified. The associated record is read into the memory buffer and the field windows on the screen will be filled with the appropriate data.

USAGE:

```
find first rec using field FIELDNAME on error goto LABEL
find last rec using field FIELDNAME on error goto LABEL
```

Begin by entering the name of the field on which to search. If the specified field is not an index-key field, an error message will be displayed to the operator during program execution. An error message will also be displayed to the operator when an attempt is made to search past the beginning or end of the file. The operator must press **[RETURN]** and then continue.

Next, enter a subroutine line label to branch to in the event of an unsuccessful operator search. If an "on error goto" label is specified in this manner, program control will transfer to the labeled command line and no error message will be printed. The label specified in this command will override a TRAP "F" (File Error) command. If no label is specified, the TRAP would be activated. See TRAP "F" (File Error) in the section on TRAP commands.

To write a FIND FIRST or LAST RECORD command:

REQUIRED entries:	FIELDNAME ON WHICH TO SEARCH
OPTIONAL entries:	COMMAND LABEL TO BRANCH TO LABEL

```
+----------------------+
|   Find               |
|   Next   Previous    |
+----------------------+
```

PURPOSE:

These commands will locate the data record following or preceding the current data record. This associated record will then be read into the memory buffer and the appropriate field windows on the screen will be filled with data.

USAGE:

"Next" or "Previous" commands must be preceded by <u>any</u> of the other FIND commands to provide a point of reference for movement to the "next" or "previous" record.

```
find next rec in file FILENAME on error goto LABEL
find previous rec in file FILENAME on error goto LABEL
```

Enter the name of the FILE in which the next/previous record is to be located. The FIND command will allow you to search sequentially through a file, by the specified key field, in ascending or descending order. A "Previous" search from the beginning record will display an **ENCOUNTERED BEGINNING OF FILE** error message. A "Next" search from the last record gives **ENCOUNTERED END OF FILE.**

Next, enter the label of a subroutine to transfer program control to in the event of an unsuccessful search. If an "on error goto" label has been specified, program flow will transfer to the labeled command line and no error message will be printed. The label specified in this command will override a TRAP "F" (File Error) command. If no label is specified, the TRAP would be activated. See TRAP "F" (File Error) in the section on TRAP commands.

"Previous" and "Next" searches are dependent on an earlier FIND which "marks a place" from which the search will be conducted. There are four conditions under which a "Previous" or "Next" search will fail:

1) The "Next" or "Previous" command was not preceded by any of the other FIND commands, so no place mark has been set.

2) During program execution you have deleted the record called in by the original FIND. This clears the place marker.

3) If you clear the record number (which tells the system that you are about to save this as a new record) you will lose your place marker when the save is executed -- but not until the save is executed.

4) During program execution you have changed the value in the key field on which the search was initiated and then saved the record back to the file. This clears the place marker.

If any of the above conditions occur during program execution, this error message will be displayed: **Must SEARCH before doing NEXT/PREV.** Note that the "Must SEARCH" warning does not trigger a TRAP "F" (File Error).

To write a FIND NEXT or PREVIOUS RECORD command:

REQUIRED entries:	FILENAME TO SEARCH
OPTIONAL entries:	COMMAND LABEL TO BRANCH TO LABEL

Print

PURPOSE:

This command is used to print information on the terminal screen, printer, or disk. When selected, the following subset of functions is displayed:

At **E**rror **F**ormat line # **W**idth = **L**ines = **M**ax lines = **B**lank lines

vertical **T**ab to line **P**age eject print **C**ontrol chrs bo**X**

These options will be explained in detail on the following pages.

Print At

PURPOSE:

This command will allow you to print a "format" line or a "message" line on the terminal. A message line is a constant (usually a string of characters) while a format line is a specified line from a reporter format screen: in other words, a variable or changing line. The information in the format line can, of course, be altered under program control.

USAGE:

This command must follow a "mount report format" command. See the "Mount" command.

Specify the row and column location where you want to print the message on the terminal. These can be constants or field names. If you use field names, the values can be altered under program control which allows you to change the print position.

If you print at column zero zero, row zero zero, the message will be displayed at the bottom of your screen and the operator must press **[RETURN]** to clear the message and continue. This allows you to stop processing, to be sure that your message is noticed. Any other column and row values will print on the screen in that location and processing is not stopped. You must move blanks into the displayed area to clear the message from the screen when using a location other than column zero zero, row zero zero.

EXAMPLES:

Message line --

THIS IS A MESSAGE]

The right bracket,] , may be used to indicate the end of the line. Only the positions (including spaces) placed to the left of the] will be printed to the terminal. The system also checks for the last non-space literal for the end of a line if the] isn't used.

Format line --

```
mount report format SEEIT print on crt
print at col 01 row ROWHOLD format line #02
```

The format line "Print at" option allows the programmer to create multiple output lines on the screen at a specified row/column location. In the example above, reporter data from line number 2 of the reporter format screen SEEIT will be printed on the screen at column number 1. Other lines in the program would be used to increment ROWHOLD so that the report will print out down the screen, one line after the other.

The line number portion of the "Print at" command line must always reference the last "mounted" reporter format screen in the program.

To write a PRINT AT command:

REQUIRED entries:	COLUMN LOCATION ROW LOCATION MESSAGE LINE OR FORMAT LINE NUMBER
OPTIONAL entries:	COLUMN LOCATION--FIELD: FIELD NAME (for format line) ROW LOCATION--FIELD: FIELD NAME (for format line) LABEL

Print Error

PURPOSE:

Use this command to print an error message at the bottom of the screen. The operator must press **[RETURN]** in order to continue with the program. Program flow will transfer to the labeled command specified.

USAGE:

Standard error messages are contained in the data file **ERRFLE.MS.** Enter the number of the error message you want to print. A list of the system error messages is provided in the Appendix of this reference manual. The number of the error message is the record number. Enter the label of the command line you want to branch to after the message is printed.

A utility program is provided that allows the system designer to add his own error messages to this data file. To add messages to the ERRFILE, execute the program ERRENT.RUN by typing "SENSIBLE ERRENT" at the operating system level, or by selecting "Main Menu 1) Execute A SENSIBLE SOLUTION Program" and specifying the program ERRENT. The record number is displayed when the user creates a new message using ERRENT.

For occasional errors and warnings, the designer should **not** add new messages to **ERRFLE**, but instead use PRINT -- at col 00 row 00 (MESSAGE).

To write a PRINT ERROR command:

REQUIRED entries:	ERROR MESSAGE NUMBER COMMAND LABEL TO BRANCH TO
OPTIONAL entry:	LABEL

```
+---------------------+
|    Print            |
|    Format line #    |
+---------------------+
```

PURPOSE:

This command allows you to print a reporter format line that was created with SCREEN PAINTING.

NOTE! If the 127th column of a format line contains a "," the next line down on the format screen will be appended. This gives you the option to print a line 255 columns wide.

USAGE:

After selecting this option, specify which format line is to be printed. The format line can be a field or a constant. Enter either the format line number or the field name where you have loaded the format line you want. If you name a field, the value in the field can be altered under program control.

Note! If the 127th position of a format line contains a comma " , " the next line down on the format screen will be appended to that line. That is, the next line becomes characters 127 through 255. This gives you the option to print a single line that is 255 characters long.

This command must follow a "mount report format" command. See the "Mount" command.

To write a PRINT FORMAT command:

REQUIRED entry: FORMAT LINE NUMBER OR FIELD NAME

OPTIONAL entry: LABEL

```
+--------------------+
|       Print        |
|       Width =      |
+--------------------+
```

PURPOSE:

Use this command to specify the width in characters of the print line.

USAGE:

After choosing this option, specify the number of characters you want in a print line. The default setting used by the system is whatever has been entered during the installation procedure. See SENSETUP in the Installation Manual. The maximum number of characters that you may print in a line is 255.

To write a PRINT WIDTH = command:

REQUIRED entry: NUMBER OF CHARACTERS WANTED IN PRINT LINE

OPTIONAL entry: LABEL

```
+----------------------+
|        Print         |
|        Lines =       |
+----------------------+
```

PURPOSE:

This command is used to specify the number of lines that you wish to print on each page.

USAGE:

Enter the number of printable lines. Report type programs will automatically page advance when this number of lines has been printed on the page. The default value used by the system is whatever has been specified during the installation procedure. See SENSETUP in the Installation Manual.

You must set a TRAP "P" (PAGE BREAK) to enable the automatic page advance. If no page advance is desired, do no set a trap.

To write a PRINT LINES = command:

REQUIRED entry: NUMBER OR PRINTABLE LINES

OPTIONAL entry: LABEL

```
+----------------------+
|      Print           |
|      Max lines =     |
+----------------------+
```

PURPOSE:

Use this command to specify the total number of lines per page. The number entered is used to calculate the amount of paper to advance between page breaks.

USAGE:

Enter the maximum number of lines available on the page. The default value used by the system is whatever has been entered during the system installation procedure. See SENSETUP in the Installation Manual.

To write a PRINT MAX LINES = command:

REQUIRED entry: MAXIMUM LINES DESIRED PER PAGE

OPTIONAL entry: LABEL

Print Blank lines

PURPOSE:

Executing this command will immediately advance the paper in the printer the specified number of lines.

USAGE:

Enter the number of blank lines you want the paper to advance between print lines.

To write a PRINT BLANK LINES command:

REQUIRED entry:	NUMBER OF LINES DESIRED BETWEEN PRINT LINES
OPTIONAL entry:	LABEL

Print vertical Tab to line

PURPOSE:

Executing this command will immediately advance the paper in the printer to the specified line number.

USAGE:

Enter the line number to which you want the paper in the printer to advance.

To write a PRINT VERTICAL TAB TO LINE command:

REQUIRED entry: LINE NUMBER TO WHICH PRINTER MUST ADVANCE PAPER

OPTIONAL entry: LABEL

```
+---------------------+
|      Print          |
|      Page eject     |
+---------------------+
```

PURPOSE:

When executed, this command will immediately force a page advance.

USAGE:

Use this command to force a page advance under program control. The system will automatically page advance, based on the number of lines the user has said are to be printed on each page, but the user can also force an advance wherever necessary.

For reports sent to the terminal, you should add this command to your "end of file" routine before exiting the program. This will have the effect of locking the screen display until the user presses the **[RETURN]** key. Without this command, the program would exit immediately and the user would not have the opportunity to view the information.

To write a PRINT PAGE EJECT command simply select the option:

REQUIRED entry: NONE

OPTIONAL entry: LABEL

```
+---------------------+
| Print               |
| print Control chrs  |
+---------------------+
```

PURPOSE:

Many terminals and printers are capable of producing unique displays such as reverse video, half intensity, compressed print, boldface, etc.. This command will allow you to send the appropriate control characters to the printer or terminal to activate such features.

USAGE:

Specify the control character sequence you wish to send to the terminal or printer. The control character sequences must be created in SENSETUP. There are 32 unique sequences available. See SENSETUP in the Installation Manual.

For example, you might use this command to change the font on the printer, direct it to utilize a 255 character print line, or send a control character to the terminal where an OCR wand or Bar Code reader has been attached

The direction of the control characters is dependent upon the currently identified output device. This is set by "MOUNTING" a reporter format screen and specifying where the output is to be directed (terminal, printer, or disk.) If no format has been mounted, direction is always to the terminal. If a format is mounted and no direction set (user defines at run time) the direction will be a result of the user selected option. You may remount reporter screens as often as you wish to re-identify the output device.

For more information, see the "Mount" command.

To write a PRINT CONTROL CHARACTERS command:

REQUIRED entry: SPECIFY CONTROL CHARACTER STRING TO BE SENT TO TERMINAL OR SCREEN

OPTIONAL entry: LABEL

```
+----------------------+
|        Print         |
|        boX           |
+----------------------+
```

PURPOSE:

This command allows you to draw boxes on the screen with specified width and depth.

USAGE:

When you select this option you are asked "AT" what location you wish to print the box. Enter the row and column locations. These can be identified as constants, or values can be passed from a field. If the user selects a field value, the values can be changed under program control.

You must also enter:

Depth = This can be a constant or field name. If a field name is used, the value in the field can be altered under program control.

Width = This can be a constant or field name. If a field name is used, the value in the field can be altered under program control.

Clear/Draw -- Specify if you want a box printed or removed from display.

With this capability, the values of calculations can be stored in fields then fed to the parameters of this command to draw bar charts on the screen.

To write a PRINT BOX command:

REQUIRED entries:	COLUMN LOCATION--CONSTANT OR FIELDNAME ROW LOCATION--CONSTANT OR FIELDNAME DEPTH--CONSTANT OR FIELDNAME WIDTH--CONSTANT OR FIELDNAME SPECIFY DRAW OR CLEAR BOX
OPTIONAL entry:	LABEL

Trap

Goto's:

- Save
- Delete
- Clear file buffer
- Exit
- Jump Screen
- File error
- Locks
- Up arrow
- down Arrow

Gosub's:

- Help
- Relates
- Page break

PURPOSE:

The TRAP command is one of the most important commands in The SENSIBLE SOLUTION. It controls program flow when conditions "outside" the program occur, such as the user pressing a control key or the occurance of an error condition. Nothing "happens" when the command is executed; instead, the command sets up what **will** happen when the specified condition occurs later during program execution.

This command sets a TRAP for a specified condition. When that condition occurs during later execution of the program, program control will be transferred to the line label specified in the TRAP command. A "Return" command must follow at the end of the subroutine called by Gosub traps.

USAGE:

Three of the TRAP commands, **relates**, **page break**, and **help** are **GOSUB**s, and the program must execute a "RETURN" command after completing the commands which are triggered. Any search triggers the "R" (relates) TRAP. When the search is completed, the "relates" subroutine is called (usually to retrieve appropriate records from related data files). Finally, command execution must return to continue from the point where the search was done. You might think of these TRAPS as "get more information and continue" activities.

The other groups are **GOTOs**; they transfer control unconditionally to the target label. **Save, delete, clear screen, file error, jump screen, exit, locks, up arrow, and down arrow,** are "finish and go on to something else" activities.

TRAPS may be turned on or off, or ignored, and can be changed anywhere in the program.

If individual commands have labels to branch to on error conditions, they will override the TRAP at execution time. (e.g., You have set a "File error" trap branch to LABEL 1. You have 2 "Find record" commands. One of them has an "on error goto LABEL 2." The other has no error label branch. The first "Find" will branch to "LABEL 2" on any error. The second "Find" will branch to "LABEL 1" as identified by the Global TRAP.

When another program is run, all TRAPS are turned off and the defaults will be used.

To create a TRAP command line in your program begin by entering a single letter to indicate one of the following TRAP types:

-- GOTO group --

S **Save --** Triggered when **[ESC]** [S] is pressed.

D **Delete --** Triggered when **[ESC]** [R] is pressed.

C **Clear file buffer --** Triggered when **[ESC]** [C] is pressed.

E **Exit --** Triggered when **[ESC]** [Q] is pressed.

J **Jump Screen --** Triggered when **[ESC]** [J] is pressed.

F **File Error --** Triggered when any file error (**End Of File**, **Beginning Of File**, etc.) occurs during execution of the program.

L **Locks --** Triggered when a user requests a record or file that has been locked . This particular command is to be used on programs that will be run on multi-user computer systems. TRAP "L" (Locks) must always be used in conjunction with either "LOCK file by Record" or "LOCK file All records."

UP ARROW -- Triggered when the **[UP ARROW]** key is pressed. Used to keep the operator from backing up the screen where inappropriate. You can set a trap for up arrow and direct the system to a specified input field or else force the cursor to stay in the current input field window.

DOWN ARROW -- Triggered when the **[DOWN ARROW]** key is pressed. Used to keep the operator from jumping ahead on the screen where inappropriate. You can set a trap for down arrow and direct the system to a specified input field or else force the cursor to stay in the current input field window.

-- GOSUB group --

H Help Screen -- GOSUB triggered when **(HELP) [ESC] [?]** is pressed.

R Relates -- GOSUB triggered when any kind of search (Find, find Beginning record, find Next record, etc.) is done by the operator or under program control.

P Page Break -- GOSUB triggered when the counter field equals the values specified by the commands "LINES =" and "MAX LINES =". The default values are set during the installation process with SENSETUP.

The options for the TRAP are:

Ignore Default Goto or sub

* IGNORE means that the TRAP function is non-existent. No action on this particular TRAP will occur. Setting to ignore causes the TRAP to be ignored and the program flow will drop through to the next sequential command.

* DEFAULT means the TRAP is unspecified. The following default conditions will occur:

File error -- Error message displayed (user press

	[RETURN] to continue)
[ESC] Quit Screen --	SENSIBLE runs MENU.RUN
[ESC] Clear Fields --	Clears all currently opened file buffers and the terminal screen
All others --	No action

Setting a TRAP to the default value causes the standard system default to occur. This will print out the standard error messages where appropriate. Where there are no defaults, such as up arrow and down arrow, no action takes place and program flow will drop through to the next sequential command.

* GOTO or GOSUB refers to the label of the command to which program flow will branch when this Trap is activated. Setting to a label causes program flow to branch to the specified label when the trap is activated. For example, if you wish to branch to a help screen for a field you have named, you can give the label of the command in your program that displays the help screen. Subsequently, this help screen will always appear when the **[?]** key is pressed. If you have multiple help screens in your program, you must set the help screen labels appropriately. You can also disable any previously set help screens by inserting a TRAP command for HELP and setting the option to ignore.

To write a TRAP command:

REQUIRED entries: DEFINE WHAT TO TRAP
SPECIFY OPTION--IGNORE, DEFAULT, OR GOTO OR SUB

OPTIONAL entries: COMMAND LABEL TO BRANCH TO
LABEL

```
+----------------------+
|   Execute            |
|   Com file           |
+----------------------+
```

PURPOSE:

Using the **execute** command you may write a command line to "execute Com file" (filename.COM). This command will allow the user to chain to an executable .COM file.

USAGE:

This command exits SENSIBLE SOLUTION, closes any open data files, clears the GOSUB stack, and then executes a specified program file (filename.COM). It is not possible to pass command-line arguments to the program being initiated.

Enter the name of the program to be executed. Specify the file **name** but not the **extension** (type) -- for example, SENSIBLE, not **SENSIBLE.COM.** This command is used within the SENSIBLE SOLUTION Menu program to execute each of the different language modules. It may also be used to invoke "alien" programs, such as operating-system utilities.

Execute
Run file

PURPOSE:

Using this command you may write a command line to "execute Run file" (filename.RUN). This command will allow you to chain to an executable SENSIBLE SOLUTION program. "execute Run file" closes any open data files, clears the **GOSUB** stack, and then initiates the specified SENSIBLE SOLUTION program.

USAGE:

Enter the name of the program to be executed (name only -- do not use the **.RUN** extension).

The command does not "mark its place" in the calling program. That is, one program cannot GOSUB to another, then return and follow its sequential command line order.

To write an EXECUTE command:

REQUIRED entry: .COM or .RUN FILE NAME

OPTIONAL entry: LABEL

```
+---------------------+
|      ! (remark)     |
|                     |
+---------------------+
```

PURPOSE:

Use the Remark command to add remarks to the source code file for program reference. The Remark command will have no effect on the execution of the program.

USAGE:

Enter whatever remark you wish wherever you want to identify the various portions of your source code file.

To write a REMARK:

REQUIRED entry: ENTER WHATEVER REMARK YOU WISH

OPTIONAL entry: LABEL

Lock

PURPOSE:

The Lock command will allow you to lock the screen, data file, or records in a data file.

USAGE:

When you select the option **L**ock by pressing [L] or highlighting it and pressing [**RETURN**], the following options will appear on your screen:

Screen   file by Record   file All records

These options will be explained in detail on the following pages.

```
+---------------------+
|        Lock         |
|        Screen       |
+---------------------+
```

PURPOSE:

This command will lock the screen with the existing data being displayed and will not refresh with subsequent data retrieval. It is typically used when a program is paging through a file looking for a specific match and the records being paged might be confusing to the operator.

USAGE:

This command should be set just prior to the "find" command. Once the "find" is accomplished, the command "Unlock screen" will refresh the screen with the current record information.

To write a LOCK SCREEN command simply select this option:

REQUIRED entry: NONE

OPTIONAL entry: LABEL

```
+-----------------------+
|    Lock               |
|    file by Record     |
+-----------------------+
```

PURPOSE:

This command will lock the record being read, making it unavailable for a read from another program. "Lock file by Record" was designed to maintain data integrity on a multi-user computer system. If you are writing programs that will be used in a multi-user environment -- a situation where more than one user may be attempting to access the same data file at the same time -- always use this command in your programs. If the system has been installed as a single-user system (see SENSETUP in Installation Manual), this command will be ignored at run-time.

USAGE:

This command should be set at the beginning of your source program for every individual file in which you want to lock records. From then on, any time that your SENSIBLE SOLUTION program reads that file it will test to see if the record is already locked; if not, this "Lock file by Record" command will lock out any other user requesting the record.

lock file (FILENAME) by record if file locked goto (LABEL)

To create a "lock file by record" command line, begin by entering the name of the file that will be read. Next you will have the option to enter the command line label to which program flow will branch if your program tries to access a file that has been locked by another user.

If the desired record is locked, the user requesting it will not be able to access that record and will be stopped from further processing until the record has been released.

If the "lock file by record" is not used in a multi-user program, no test will be made to check for locked records, and all reads and saves will be executed as requested. In a program designed to only read a data file, you may wish to refrain from locking records because it will allow the user to read any record in the specified data file whether it is locked or not.

If a "Trap 'L' (locks)" command is also used in your program, any time that a user requests a locked record the trap will be activated and program flow will branch to the line label specified in that command line.

By clearing the record number field with a "clear record number" command, the system will unlock this particular record. This allows you to work with data in the record while making it available to other users. This is typically done in applications where you have a system file usually consisting of one record that contains information that other programs use (date, company name, invoice numbers, etc.). If you are updating this record, you need to read it in again "LOCKED", update the information, and write it back out.

A record will be automatically unlocked when you save the record back to the disk and clear the buffer, page through the file, search for another record, or perform a clear buffer command.

To write a LOCK FILE BY RECORD command:

REQUIRED entry:	FILE NAME BEING READ
OPTIONAL entries:	COMMAND LABEL TO BRANCH TO LABEL

```
+----------------------+
|   Lock               |
|   file All records   |
+----------------------+
```

PURPOSE:

This command will lock an entire file and make it inaccessable to other users provided that the other user's programs also employ a "lock file all records" command. This command was designed to maintain data integrity on a multi-user computer system. If you are writing programs that will be used in a multi-user environment -- a situation where more than one user may be attempting to access the same data file at the same time -- always use this command in your programs. If the system has been installed as a single-user system (see SENSETUP in Installation Manual), this command will be ignored at run time.

USAGE:

lock file (FILENAME) all records if file locked goto (LABEL)

This command should be placed at the beginning of any program you write that will be used in a multi-user environment. Before locking all of the records in a file, this command will first test to see if a record or all of the records are already locked in the file. If either one of these two lock conditions is sensed, the command will route program flow to a previously specified line label and no lock will be placed on the file.

To create a "lock file all records" command, enter the name of the file to be locked. Next you have the option of entering a command line label to which program flow will branch should the program attempt to lock a file that is already locked.

A user request to a locked file will halt processing for that user unless a label is specified in the "lock file all records" command line. A file can be unlocked under program control by executing an "Unlock file" command. Any time that the user exits the program, the files that were previously specified as "locked" will automatically be unlocked.

If the "lock file all records" is not used in a multi-user program, no test will be made to check for locked records, and all reads and saves will be executed as requested. In a program designed to only read a data file (not to write to the data file), you may wish to refrain from locking records because it will allow the user to read any record in the specified data file whether it is locked or not.

If the system has been setup as single-user when running SENSETUP, this command will be ignored. (See SENSETUP in the Installation Manual.)

To write a LOCK FILE ALL RECORDS command:

REQUIRED entry: FILE NAME TO BE LOCKED

OPTIONAL entries: COMMAND LABEL TO BRANCH TO
LABEL

Unlock Screen

PURPOSE:

This command can be used to unlock a screen on a multi-user system.

USAGE:

unlock screen

Once the requested record information has been found, this command will unlock the screen and refresh it with the data.

To write an UNLOCK command:

REQUIRED entry:	SCREEN NAME
OPTIONAL entry:	LABEL

Unlock File

PURPOSE:

This command unlocks the file specified. It does not affect locks set by other programs .

USAGE:

unlock file (FILENAME)

After selecting this option, name the file to be unlocked. This command is used to release file locks set by the issuing program only. It typically is used to avoid "fatal embrace" occurrences where two programs have locked records that each is requesting of the other.

With a TRAP "L" (Locks) command the programmer could unlock all files of a program, then reset the record locks and try again to access the specific record being searched for. See TRAP "L" (Locks) command.

If the system has been setup as single user when running SENSETUP, this command will be ignored. (See SENSETUP in the Installation Manual.)

To write an UNLOCK command:

REQUIRED entry:	FILE NAME
OPTIONAL entry:	LABEL

I
RECORD LAYOUT

Records created with the SENSIBLE SOLUTION are stored in the file in sequential order. The values stored in fields defined as keys are also kept in a separate key file along with the record number they come from . When a search is performed, the system first looks to the key file for the value being searched. When located in the key file, the record number for the particular value is extracted and then the actual data record is retrieved from the data file.

The record layout is determined by the fields that are created for each data file. The field names for each record are sorted into alphabetical order and, as the record is built, the fields will be placed in this sequence. The INITIALIZE utility performs this task of sorting the variables into alphabetical order.

No delimiters are inserted between fields. The system tracks the offset within the record that identifies the beginning position of each field and the length.

Records are built in multiples of from 128 characters with no limit on the total number or characters per record. All of the data in a record is stored as ASCII characters. These records can be read by most other languages.

A SENSIBLE SOLUTION data file comprises two actual files on disk: a master-data file, filename.MS, and an associated index-key file, filename.KS. The key file, filename.KS, is automatically generated and maintained by SENSIBLE SOLUTION. The programmer need not be concerned with its internal structure.

The master-data file, filename.MS, is a non-delimited, fixed-length-field record structure. All data is kept in ASCII form. No nulls or "control characters" are stored in the file; in particular, a record does not end with a <CR>/<LF> pair. Fields within a record are defined solely by their position; they are not delimited by commas or quotes.

All disk I/O is done in blocks of 128 bytes; if necessary, the physical record size is rounded up to the next larger multiple of 128. Any unused bytes in the last block are filled with blanks (ASCII decimal 32) and ignored. This blocking prevents "cross-boundary," record-locking problems in multi-user systems.

Each field in a record is defined in the Data Dictionary by field name, data type, field width, decimal places (numeric type only), and index-key status. Fields are stored in the record in alphabetical order by field name. The starting position ("offset") for each field is determined by the width of the fields preceding it in field-name order.

Example: a data file CUSTOMER.MS has the following fields:

Field Name	Type	Size	Decimals	Key?	Value
CUST.NAME	Alpha	20		Y	SKYWALKER, LUKE
CUST.DATE	Date	6		N	08/26/83
CUST.PHONE	Alpha	12		Y	909/555-1212
CUST.BILL	Numeric	10	2	N	1024.45

Total record width =		48			

A data record using the definition specified above would look like this (bytes 128-48 = 80 blank filled):

```
         1         2         3         4         5
0...v....0....v....0....v....0....v....0....v....0 --
   1024.45 30585SKYWALKER, LUKE     909/555-1212
                                    ------------CUST.PHONE
                ----------------------------CUST.NAME
          ------CUST.DATE
----------CUST.BILL
```

A final point should be made here concerning deleted records. Any time a record is deleted from a file, SENSIBLE SOLUTION will pad the entire record with blanks. This record of blanks is then automatically re-used by SENSIBLE SOLUTION on a "last deleted, first to be re-used basis."

II
FIELD TYPES

The SENSIBLE SOLUTION will handle 5 different types of field definitions -- A (alphabetic), N (numeric), D (date), O (overlay), and R (record).

Alpha: An alphabetic type field will store printing character ASCII data from ASCII decimal 32 to ASCII decimal 126. Entries shorter than the full field width are blank-filled on the right.

Numeric: Numeric ASCII data, including leading minus sign and embedded decimal point if appropriate. Numbers will be stored right-justified with the correct number of decimal places (e.g. **-123.000**, not just **-123**).

Date: ASCII Julian number. Dates can be entered in American short or long form, mm/dd/yy or mm/dd/yyyy; or dates may be entered in European short or long form, dd/mm/yy or dd/mm/yyyy. All date forms are converted internally to a "days since base date" form. This allows addition ("What is the date 10 days after 3/10/83?"), subtraction ("How many days between 12/31/80 and 3/15/81?"), and sorting by date. To specify the type of date that you want your system to use, you must use SENSETUP.COM (see the Installation Manual).

Overlay: An overlay field is a field that combines all or a portion of several other fields into one field. Let's look at the diagram below:

```
0        9          20             35             50
|--------+----------+--------------+--------------+---------
|field A | field B  |   field C    |    field D   | field E
|--------+----------+--------------+--------------+---------
                       ^                    ^
                       |                    |
                       |                    |
                       |                    |
                       +----Overlay Field---+
```

In this example the overlay field of 17 characters spans two fields -- field C and field D. You do not have to use all of the fields to create an overlay. Notice that in our example that the overlay begins at location 23 and ends at location 44 when, in fact, fields C and D begin at location 20 and end at location 50.

To create an overlay field that would match our example, begin by specifying "O" in the "Type" field window of the Data Dictionary screen. The cursor will move into the next field window, "Size," where you should specify the length of the overlay field -- in this case, 17 characters. The cursor will then skip the next field window, "Decimal," and move to "Offset." At this field window you must type in the location of the beginning of the overlay field -- 23. Finally, specify whether or not the overlay field will be a key field.

Overlay fields can not only be used to combine other smaller fields into a larger field, they can also be used to break apart one very large field into two or more smaller and more manageable overlay type fields. For example, since your terminal is probably only 80 characters wide you may have considerable difficulty creating a program that will accept a 255 character input field -- the maximum allowable size of a field. However, by using overlay fields you can break the desired field into four smaller overlay type fields consisting of three fields of 80 characters and one field of 15 characters. Data can then be entered into the four smaller overlay fields and they will be automatically combined into one large 255 character field.

The subject of overlay fields brings up an interesting problem. Since overlay fields will require that two or more fields adjoin each other in a particular order, you may begin to wonder how you can easily control the position of data fields within a record. Remember that the SENSIBLE SOLUTION arranges all fields in alphabetical/ASCII code order so that field names that begin with an "A" would precede the location of field names that begin with a "B," etc.. With this in mind, the most obvious solution to the problem is to adapt a field naming convention that will enable you to arrange the fields within the record as you like.

The Data Dictionary provides a field window called "Field Description" to help identify the field. We recommend that you use this instead of the field name to identify the field. By doing this, you can use more abstract names for fields and thus concentrate on manipulating the position of the fields. Overlay fields are one 'of the more powerful features of the SENSIBLE SOLUTION. You can go so far as to create overlay fields of overlay fields, of overlay fields, ad infinitum. However, there are some restrictions to using overlay fields that you should keep in mind:

-- It is impossible to overlay an "R type" (record) field because R type fields have a zero offset.

-- You should not overlay a portion of a date field. Generally speaking you must overlay the entire field if it is a date. The SENSIBLE SOLUTION maintains the date internally as an 8 digit number representing the number of days from January 1st., A.D. 0000.

Record: A record type field is a field that acts as a record pointer within a file. Any time that a value is placed in the "R" type field, SENSIBLE SOLUTION will immediately find and retrieve the record whose number matches that value. In this sense an "R" type field always behaves like a key field because it forces a search and retrieve operation.

Because the field does not actually take up physical space on the disk, it always has a zero "Offset" value. However, if you add an "R" type field to an existing data file, you must INITIALIZE and REKEY the file just as you would with any field that occupies physical disk space.

If you load a record number that does not exist into an "R" type field, you will get a SENSIBLE SOLUTION disk read error.

Before we leave this section on field types, we should discuss the use of internal memory variables. As you know, SENSIBLE SOLUTION views all variables as fields -- fields that occupy a physical space in disk memory. Frequently, though, programmers need to use temporary variables to store things like accumulators or counters. These kinds of variables will never require permanent storage on the disk. With the SENSIBLE SOLUTION you can do this by creating a special data file that we call MEMORY.MS/.KS. The actual disk file will consist of only one empty record subdivided into all of the field (variable) definitions that you may require.

Although we never actually write to this file on the disk, it must, nonetheless, be created and INITIALIZED -- just like any other disk file. The file does not have to be called MEMORY. You can call it anything you like. The important thing is that you should create a file that is strictly reserved for holding temporary variables. And, use it just for that one purpose -- don't ever save a record in this special temporary variable file.

III
ALTERING A DATA STRUCTURE

The structure of fields and records in a SENSIBLE SOLUTION program is controlled by the Data Dictionary. If you make changes in the Data Dictionary to a record or field definition utilized by a SENSIBLE SOLUTION program, you will have to follow several specific procedures to maintain a usable data structure.

Any time that you change a field definition, every SENSIBLE SOLUTION program that accesses that field must be re-compiled by using main menu selection 6, "Compile A Source Code File" (SENSCOMP.COM). Changing a field definition would include any change that you make to the following field windows on the Data Dictionary screen: "Type," "Size," "Decimal," "Offset, or "Key".

If the programs that access that field have already been used to generate data files, you must perform a specific procedure to maintain the validity of your data files. If you do not follow the correct procedure, your data file(s) may become hopelessly corrupted and the data essentially lost. Look at the logic chart below and determine which set of conditions applies to your particular situation. We strongly recommend that you make back-up data files and re-name them before performing any of the following procedures.

	No Data In Data Files	"Live" Data In Data Files
You have changed Type, Size, Decimal, or Offset of a non-key field	INITIALIZE COMPILE	RESTRUCTURE COMPILE
You have changed Size, Key (Y/N), or added a new field	INITIALIZE COMPILE	RESTRUCTURE REKEY COMPILE

Procedure	Menu Selection #	Description	Program
INITIALIZE	5	Initialize A Data File	SENSCOMP.COM
COMPILE	6	Compile A Source Code File	SENSCOMP.COM
REKEY	7	Rekey A Data File	SENSRKEY.COM
RESTRUCTURE	8	Restructure A Data File	SENSRSTC.COM

IV
TRANSLATING FOREIGN DATA

If you would like to use SENSIBLE SOLUTION to manipulate data from another database, you must insure that the data is in a format that SENSIBLE SOLUTION can read. In other words, if a database is not compatible with the SENSIBLE SOLUTION database, you will be required to translate the data into the SENSIBLE SOLUTION record structure. To do this, you will need to write a program in a language other than SENSIBLE SOLUTION such as BASIC, Pascal, Assembler, etc..

Transforming Foreign Files Into SENSIBLE SOLUTION Format:

To move data from an outside application to SENSIBLE SOLUTION, you must enter the appropriate field definitions in the Data Dictionary, create filename.MS, and REKEY the file to construct filename.KS. Use the following procedure:

Step 1 -- Define your field specifications (Field name, Type, Size, Decimal, Key) and enter them in the Data Dictionary. You may use main menu selection 2, "Data Dictionary Maintenance," or create a file maintenance program of all of your target fields with main menu 3, "Screen Painting" and main menu 9, "Program Generator." The file maintenance program will allow you to inspect the newly-transported data file for accuracy. Remember that the Data Dictionary will automatically arrange the fields within the record in alphabetical field-name order. This order cannot be overridden.

Step 2 -- If you entered the field definitions directly through the Data Dictionary, remember to INITIALIZE the data file. If you use the second method, "Screen Painting" and "Program Generator," the data files will be automatically INITIALIZED. Either method will create an "empty" pair of files -- filename.MS/.KS.

Step 3 -- Erase filename.MS, it will be replaced by the file you are about to generate. Do not erase filename.KS.

Step 4 -- Generate a new filename.**MS** with an appropriate BASIC, Pascal or other program. The data must be of the correct "Size" (width) for each field and in the position within the record specified by the Data Dictionary (alphabetical field order by field name).

You should note that the physical record size must be a multiple of 128 bytes and no larger than 1536 bytes (12 128-byte blocks). Fill the "unused" bytes, if any, with blanks. The record should not end with a <CR>/<LF> pair.

Dates in Gregorian format (mm/dd/yy, dd/mm/yy, etc.) cannot be moved directly into a date-type field. To convert a date from Gregorian format to Julian format (the internal form), you must move the Gregorian string (mm/dd/yy, etc.) to an 8 or 10 character alpha field. Next, write a SENSIBLE SOLUTION program that moves the alpha field to a new "D" type (date) field. The SENSIBLE SOLUTION will perform the conversion to internal Julian form automatically.

Step 5 -- REKEY the data file.

Step 6 -- Both of the files, filename.MS and .KS, are now valid SENSIBLE

SOLUTION data files. You may now write SENSIBLE SOLUTION programs to perform any further transformations required.

Transforming SENSIBLE SOLUTION Files Into Other Formats:

Information contained in SENSIBLE SOLUTION data files may also be accessible by alien applications. By using main menu 10, "Inquire," or by writing a special report generating SENSIBLE SOLUTION program, you can translate your current data structure into a form that is easily accessable.

"Inquire" is an an ad-hoc "quick query" facility that generates a listing of some or all fields from a single data file. The records selected for listing may be controlled by conditional testing of the contents.

It is also an easy matter to write a SENSIBLE SOLUTION program that will control the output format to create delimiters, headings, footings, etc. as you like.

Using either the "Inquire" program or a special report generating program will limit you to the maximum width of your line printer as specified in the system installation program, SENSETUP.COM. The maximum allowable width of any SENSIBLE SOLUTION report is 255 characters. You may, however, send larger data sets in multiple-line groups. The "Inquire" program and a report generating program can send a print out of a data file to the CRT screen, the line printer, or to a disk file. The disk file is a "print-image" file that includes <CR>/<LF> pairs at the end of each line plus page breaks and headings.

Whichever method you choose, fields will always be sent full-width, including leading or trailing blanks. It is your responsibility to deal with blanks, delimiters, and so on from within the alien application.

V
MULTI-USER CONSIDERATIONS

Record Locking and File Locking:

The multi-user version of SENSIBLE SOLUTION supports both file locking and record locking. This means that the language, and all programs written in the language, has complete multi-user capabilities. The SENSIBLE SOLUTION can be purchased as either a single user type system or as a multi-user type system. To get multi-user capabilities from SENSIBLE SOLUTION you must have purchased the multi-user version and you must have specified multi-user setup when the system was installed with SENSETUP.COM (see the Installtion Manual).

When file locking is invoked in a filename.RUN program, one user essentially "owns" the particular file that is being accessed. If some other user tries to access the file, that user's terminal will display a "file in use" message until it can successfully access the file. File locking is usually employed in a program when a global update on a data file is being done. For example, in an accounting application such as posting to the General Ledger, you should lock the file during the posting operation.

A thoughtful programmer will usually employ a trap in the program that will cause a message to be printed on the screen when any user encounters a lock out situation. This helps eliminate any anxiety a user might feel when their terminal "hangs up."

Record locking works much the same as file locking except that only one record is locked at any one time. If your filename.RUN program updates a record, you must assume that some other user will attempt to access that record at the same time. Do not risk corrupting the record. Lock the record that is going to be updated. Since a record can be updated in such a short time, you probably can dispense with a lock out trap and "file in use" message.

SENSFREE:

If you are using any of the language main menu selections (SENS*.COM) and you are currently accessing a file such as a source code file (filename.SRR) or a screen format file (filename.SCC), all other users will be locked out of that file until the file is saved back on the disk. The intruding user will see a "file in use" message displayed on their terminal.

Sometimes you will unavoidably end up making a "disorderly exit" from a program (SENSINIT.COM, SENSRKEY.COM, SENSRSTC.COM, etc.) that has access to the Data Dictionary. This would happen, for instance, if you accidentally pressed the [BREAK] or [RESET] key on your terminal during program execution or it could happen as a result of a power failure. The result of this is that your data files could be left open, or worse, permanently "locked".

If this condition occurs, you may well get a "Cannot gain exclusive use of Data Dictionary" error message when you try to run one of the selections from the main menu. To alleviate this condition, run the program **SENSFREE.COM** and specify the name of the data file that is locked. SENSFREE will simply turn off the "file-is-locked" flag in the specified filename.KS file. Try running one of the main menu programs again and you should have no more access problems.

Drive Locations of Data Dictionaries:

In a multi-user environment there may be a number of Data Dictionaries present on a system. This is frequently the case where a number of different users are engaged in program modification or development. For people involved in this situation, it is important to understand how SENSIBLE SOLUTION determines which Data Dictionary (RECFLE.MS/.KS and FLDFLE.MS/KS) it is going to use.

To begin, we will explain how SENSIBLE SOLUTION finds data files (including the data files that comprise the Data Dictionary) during program execution. You will recall that the installation process of SENSIBLE SOLUTION requires that you run a program called SENSETUP. One of the functions of this program is to inform SENSIBLE of the drive location of the Data Dictionary. Actually, SENSETUP really only conveys the drive location of RECFLE.MS, which is only one of four data files comprising the Data Dictionary. When the installation procedure is completed this drive location information is imbeded in a file called SENSCTRL.MS. SENSCTRL must always be present on the drive that SENSIBLE is being called from.

When you run a SENSIBLE SOLUTION program (filename.RUN) the first thing SENSIBLE does is to open SENSCTRL (on the currently logged on drive) and read the imbeded drive location of RECFLE.MS. SENSIBLE then goes to the prescribed drive location, opens RECFLE.MS, and reads the drive locations of every data file that will be accessed by that particular program. Remember that one of the functions of RECFLE.MS is to store the drive locations of all data files. The data files can be located on any drive on the system.

If a program (filename.RUN) happens to require the use of the data file RECFLE, it will search for RECFLE on the drive location specified in the first RECFLE.MS. Thus, you can see that the RECFLE.MS specified in

SENSCTRL/SENSETUP may not be the same RECFLE specified in RECFLE.MS.

Herein lies the potential for problems. Is SENSIBLE really looking at the Data Dictionary that SENSETUP indicates? Ideally, you should set the location of the Data Dictionary files (RECFLE and FLDFLE) as specified in SENSCTRL/RECFLE.MS, so that RECFLE always points to itself. That way, when you run SENSETUP again to determine the location of the Data Dictionary, the specified drive location will in fact contain the Data Dictionary you are looking for.

To review, SENSIBLE needs to know the current location of the data files that will be accessed. It gets those locations from RECFLE.MS and it gets the location of RECFLE.MS from SENSCTRL. Don't risk causing confusion for yourself or other users:

> **Set the drive location of the Data Dictionary (RECFLE/FLDFLE), as defined by SENSETUP, and the drive location specified in RECFLE.MS for finding RECFLE and FLDFLE to the same location.**

data file, a colon, and the name of the data file. Remember, that all SENSIBLE SOLUTION data files come in pairs consisting of filename.**MS** and filename.**KS**. You do not have to specify the file extensions, .MS or .KS.

SENSALOC will scan the specified disk drive for the .MS and .KS files and, if the requested file pair can not be found or accessed, it will re-prompt you for the file name. When the two files are located, SENSALOC will report their current size and then print out a projection of their size if the file pair is extended by 25%, 50%, 100% (doubled), 200%, and 300%. Next, it will ask if you wish to enlarge the file. If you choose to extend the file, SENSALOC will ask by how much. You can extend the file from 1% to 999%. Simply type in the percentage you want. Extending a file by 999% will increase its size by a factor of eleven.

Next, SENSALOC will tell you how much free disk space will be required and ask you for approval. Answer "Y" to pre-allocate the file size. One caveat is in order here; **before you pre-allocate a file, you must be certain that the file is closed and no users are attempting to open it.** Failure to take this precaution could cause the data already in the file to be seriously corrupted.

Two final points. From the discussion above, it is obvious that before a file can be enlarged it must first contain some data. A 0k file enlarged by 200% is still just a 0k file. So if you are creating a brand new file and want to pre-allocate its size, be sure that you first create some base data that SENSALOC can use as a foundation for calculating the size increase. If you need, you can run SENSALOC over and over again to create a pre-allocated file of whatever size you desire. You will only be limited by the available space on your disk drive.

MULTI.SYS On DPC/OS-like O/S's:

Some operating systems (specifically DPC/OS, Mmmost, and Network O/S) utilize a special system information file called MULTI.SYS. SENSIBLE accesses this file to determine the current file status of the operating system. For SENSIBLE to run on a DPC/OS-like operating system, MULTI.SYS must be present on the same drive location as SENSIBLE.

data file, a colon, and the name of the data file. Remember, that all SENSIBLE SOLUTION data files come in pairs consisting of filename.**MS** and filename.**KS**. You do not have to specify the file extensions, .MS or .KS.

SENSALOC will scan the specified disk drive for the .MS and .KS files and, if the requested file pair can not be found or accessed, it will re-prompt you for the file name. When the two files are located, SENSALOC will report their current size and then print out a projection of their size if the file pair is extended by 25%, 50%, 100% (doubled), 200%, and 300%. Next, it will ask if you wish to enlarge the file. If you choose to extend the file, SENSALOC will ask by how much. You can extend the file from 1% to 999%. Simply type in the percentage you want. Extending a file by 999% will increase its size by a factor of eleven.

Next, SENSALOC will tell you how much free disk space will be required and ask you for approval. Answer "Y" to pre-allocate the file size. One caveat is in order here; **before you pre-allocate a file, you must be certain that the file is closed and no users are attempting to open it.** Failure to take this precaution could cause the data already in the file to be seriously corrupted.

Two final points. From the discussion above, it is obvious that before a file can be enlarged it must first contain some data. A 0k file enlarged by 200% is still just a 0k file. So if you are creating a brand new file and want to pre-allocate its size, be sure that you first create some base data that SENSALOC can use as a foundation for calculating the size increase. If you need, you can run SENSALOC over and over again to create a pre-allocated file of whatever size you desire. You will only be limited by the available space on your disk drive.

MULTI.SYS On DPC/OS-like O/S's:

Some operating systems (specifically DPC/OS, Mmmost, and Network O/S) utilize a special system information file called MULTI.SYS. SENSIBLE accesses this file to determine the current file status of the operating system. For SENSIBLE to run on a DPC/OS-like operating system, MULTI.SYS must be present on the same drive location as SENSIBLE.

The SENSIBLE SOLUTION Language

SYSTEM SPECIFICATIONS

Maximum Program Size	O/S Limited*
Maximum Data File Size	O/S Limited*
Maximum Number of Data Files	Unlimited
Maximum Number of Records per Data File	16,777,216
Maximum Number of Data Fields per Record	1,000
Maximum Bytes per Data File Record	26,496
Maximum Number of Open Files in a Program	16
Maximum Number of Indexes (Keys) per Data File Record (This includes One Pre-Defined Record Number Index)	10
Maximum Number of Keys per Screen or Program	160
Maximum Length of Key Field	72
Maximum Length of a Single Field	255
Stored Number Range:	
Maximum	+99,999,999,999.9999
Minimum	- 9,999,999,999.9999
Decimal Place Precision (Computations are done to 5 decimal place precision, then rounded to the precision of the target field.)	4
Maximum Number of Accessed fields per Program	255**
Maximum Number of Command Lines per Program	2,000
Maximum Number of Command Line Labels per Program	300
Maximum Number of Nested Subroutines (GOSUB)	20
Maximum Length of Reporter Print Line	Printer Limited
Maximum Number of Report Format Lines	60
Maximum Fields (fields) on a Screen/Report Format	255
Maximum Length of Field (Variable) Name	15

* O/S Limited means limited by the disk capacity and operating system.

** An array is considered as a single field.

The SENSIBLE SOLUTION Language

COMPLETE FILE LIST

CP/M	CP/M-86	MS-DOS
ENTFLE .RUN	ENTFLE .RUN	ENTFLE .RUN
ERRENT .RUN	ERRENT .RUN	ERRENT .RUN
ERRFLE .KS	ERRFLE .KS	ERRFLE .KS
ERRFLE .MS	ERRFLE .MS	ERRFLE .MS
FLDFLE .KS	FLDFLE .KS	FLDFLE .KS
FLDFLE .MS	FLDFLE .MS	FLDFLE .MS
MAILLIST.KS	MAILLIST.KS	MAILLIST.KS
MAILLIST.MS	MAILLIST.MS	MAILLIST.MS
MAILLIST.RUN	MAILLIST.RUN	MAILLIST.RUN
MAILLIST.SCC	MAILLIST.SCC	MAILLIST.SCC
MAILLIST.SRR	MAILLIST.SRR	MAILLIST.SRR
FLOPMENU.RUN	FLOPMENU.RUN	FLOPMENU.RUN
MENU .RUN	MENU .RUN	MENU .RUN
MENU .SCC	MENU .SCC	MENU .SCC
MENU .SRR	MENU .SRR	MENU .SRR
RECFLE .KS	RECFLE .KS	RECFLE .KS
RECFLE .MS	RECFLE .MS	RECFLE .MS
SENSCTRL.MS	SENSCTRL.MS	SENSCTRL.MS
SENSCMD .COM	SENSCMD .CMD	SENSCMD .EXE
SENSCOMP.COM	SENSCOMP.CMD	SENSCOMP.EXE
SENSCRN .COM	SENSCRN .CMD	SENSCRN .EXE
SENSETUP.COM	SENSETUP.CMD	SENSETUP.EXE
SENSGEN .COM	SENSGEN .CMD	SENSGEN .EXE
SENSIBLE.COM	SENSIBLE.CMD	SENSIBLE.EXE
SENSINIT.COM	SENSINIT.CMD	SENSINIT.EXE
SENSINQR.COM	SENSINQR.CMD	SENSINQR.EXE
SENSRKEY.COM	SENSRKEY.CMD	SENSRKEY.EXE
SENSRSTC.COM	SENSRSTC.CMD	SENSRSTC.EXE
SENSFREE.COM	SENSFREE.CMD	SENSFREE.EXE
TERMDEFS.MS	TERMDEFS.MS	SENSMAIN.EXE
		TERMDEFS.MS

The SENSIBLE SOLUTION Language

FILE EXTENSIONS

File extensions created and used by The SENSIBLE SOLUTION Language:

filename.SCC	Screen layout source file
filename.LST	A text file of screen layout
filename.SRR	Command source code file
filename.RUN	Compiled command file
filename.MS	Master data file
filename.KS	Key file (for the .MS file)
filename.IQ	Inquire format file

Operating System file extensions used by The SENSIBLE SOLUTION:

filename.COM	Compiled executable program -- CP/M, MP/M
filename.EXE	Compiled executable program -- MS-DOS, PC-DOS
filename.CMD	Compiled executable program -- CP/M-86

The SENSIBLE SOLUTION Language

FILES

Run-Time Module Files:

SENSIBLE.COM	Main Language Executive Program
MENU .RUN	Language Selection Menu Program
SENSINQR.COM	Inquery Program for a Single File
RECFLE .MS/.KS	Data Dictionary Files of all Files
FLDFLE .MS/.KS	Data Dictionary Files of all Fields
ERRFLE .MS/.KS	Language Message Files
SENSCTRL.MS	Terminal Control Code Data File

Language Files:

ENTFLE .RUN	Dictionary Maintenance Program
SENSCRN .COM	Language Screen Editor
SENSGEN .COM	Language Program Generator
SENSCMD .COM	Language Command Source Code Editor
SENSCOMP.COM	Language Program Compiler
SENSINIT.COM	Data File Initialization Program
SENSRKEY.COM	Rebuilds Data Key Files
SENSRSTC.COM	Restructures Data Files

Language Utility Files:

SENSETUP.COM	Installation/Configuration Program
SENSFREE.COM	Multi-user Utility Program
ERRENT .RUN	Error Message Program

The SENSIBLE SOLUTION Language

Main Menu Selections

1)	SENSIBLE.COM	Execute A SENSIBLE SOLUTION Program
2)	ENTFLE .RUN	Data Dictionary Maintenance
3)	SENSCRN .COM	Screen Entry
4)	SENSCMD .COM	Source Code Editor
5)	SENSINIT.COM	Initialize A Data File
6)	SENSCOMP.COM	Compile A Source Code File
7)	SENSRKEY.COM	Rekey A Data File
8)	SENSRSTC.COM	Restructure A Data File
9)	SENSGEN .COM	Program Generator
10)	SENSINQR.COM	Inquire

The SENSIBLE SOLUTION Language

System Error Messages

1 Disk error while reading file

2 Disk error while writing file

3 Disk error while opening file

4 Disk error while closing file

5 Disk error while creating file

6 Disk error while deleting file

7 Disk error while attempting to lock

8 Disk error while attempting to unlock

9 Disk error while renaming file

10 Invalid drive specification

11 Logical record is too long

12 Too many overlay fields in record

13 Too many Fields defined in record

14 Overlay Field extends beyond the end of record

15 Overlay offset not within defined fields

16 Nine Keys have already been specified

17 Key length exceeds 72 characters

18 Field size is zero

19 Cannot Initialize RECFLE or FLDFLE

20 Specified drive conflicts with Data Dictionary definition

21 Cannot gain exclusive use of Data Dictionary

22 Data Dictionary has been updated, report spec is invalid
Please re-enter the specifications for this report

23 ---

24 ---

25 Report would exceed printer width defined in SENSETUP

26 Maximum number of fields have been selected

27 Maximum number of Criteria have been selected

28 Field is not in selected file

29 ---

30 ---

31 No source file

32 Source file to insert is null

33 Fieldname must be less than or equal to 15 characters

34 Cannot mix String and Numeric types in calc expression

35 Type must be [], () or <>

36 Operator must be one of the following: + - * /

37 ---

38 ---

39 ---

40 ---

41 Field is at right margin, cannot move right

42 Field would overlap right margin

43 Cannot remove, no field at cursor position

44 Field is already on the screen elsewhere

45 Field would overlay another field on screen

46 Screen file not found

47 ---

48 ---

49 ---

50 ---

51 Too many files referenced in .RUN file

52 More than 255 fields referenced in .RUN file

53 Too many branch labels defined

54 Duplicate branch label defined

55 Branch label not defined

56 Too many screens to compile

57 ---

58 ---

59 ---

60 ---

61 File has no records

62 File has no keys

63 File not found in Data Dictionary

64 Field not found in Data Dictionary

65 Memory exhausted, cannot continue

66 End of program reached without RETURN or GOTO

67 ---

68 ---

69 ---

70 Key not found on delete

71 Must find a record (BEGIN/END/FIND) before doing NEXT/PREV

72 Find reached beginning of keys

73 Find reached end of keys

74 Exact match not found

75 Keyfile corrupted, please rekey

76 Duplicate key already exists

77 Related record not found

78 Field is not a key, cannot search

79 ---

80 Invalid array reference

81 Arithmetic result too wide for field, data invalid

82 Arithmetic overflow during calculation, data invalid

83 Attempt to divide by zero, data invalid

84 Invalid date

85 Mask length exceeds field width

86 Field is not type 'A', cannot justify

87 ---

88 ---

89 ---

90 More than 20 GOSUBs nested

91 RETURN without GOSUB

92 .RUN file contains invalid command, please recompile

93 .RUN file not found

94 Internal error, please contact your dealer or
O'Hanlon Computer Systems at (206) 885-2502

95 Multi-user access unsuccessful

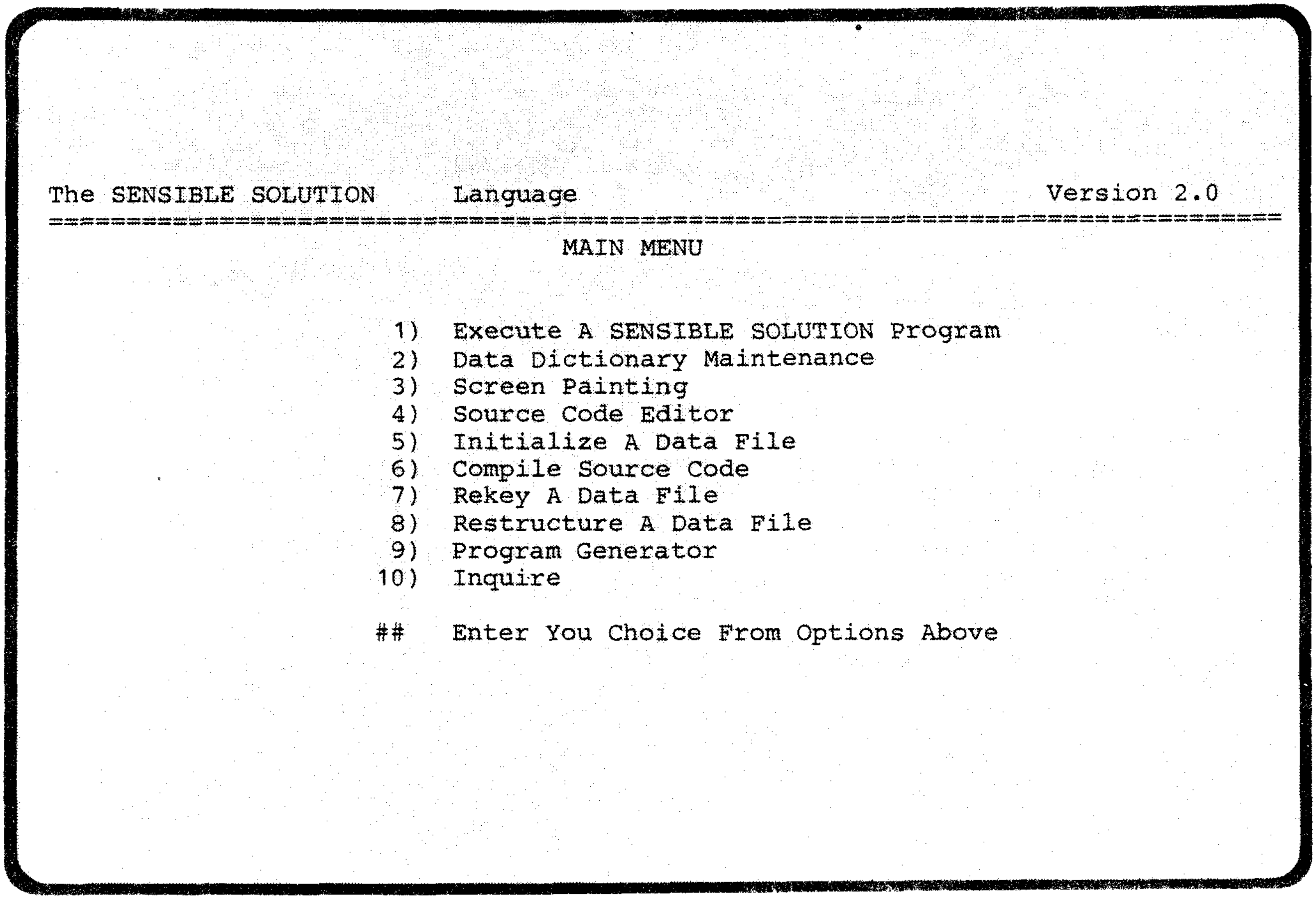
96 MULTI.SYS not found

97 Demo version, 150 record limit exceeded

98 ---

99 ---

MENU.SCC



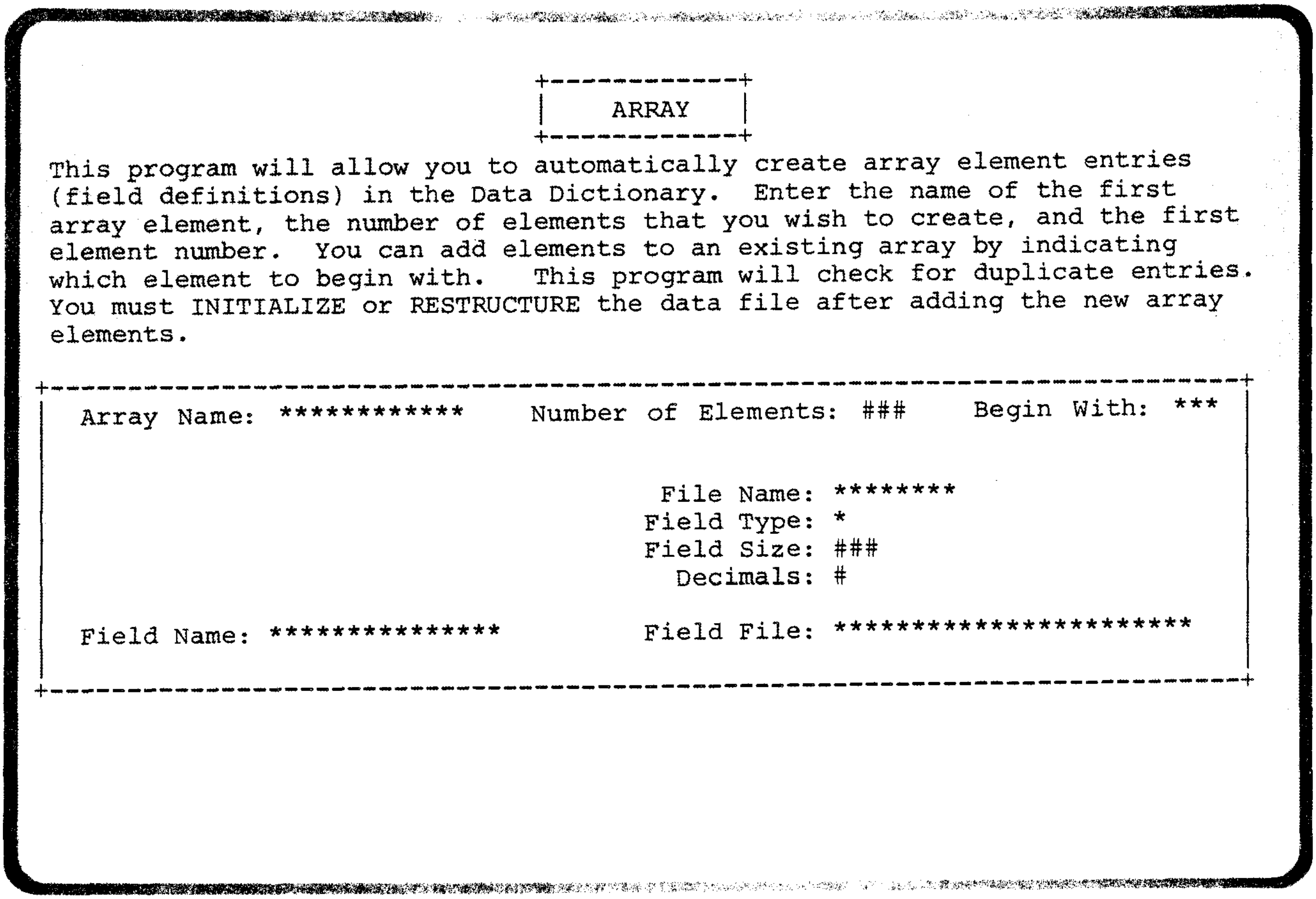
```
The SENSIBLE SOLUTION     Language                                     Version 2.0
===============================================================================
                                    MAIN MENU

                           1)   Execute A SENSIBLE SOLUTION Program
                           2)   Data Dictionary Maintenance
                           3)   Screen Painting
                           4)   Source Code Editor
                           5)   Initialize A Data File
                           6)   Compile Source Code
                           7)   Rekey A Data File
                           8)   Restructure A Data File
                           9)   Program Generator
                          10)   Inquire

                          ##    Enter You Choice From Options Above
```

Field name	File	Size	Col	Row	Key
N.2.0.1	MEMORY	2	022	17	N

MENU.SRR

```
0001           remark MAIN MENU PROGRAM FOR SENSIBLE SOLUTION LANGUAGE
0002           goto MENU
0003           remark PORTIONS COPYRIGHT 1983 O'HANLON COMPUTER SYSTEMS, INC.
0004           remark H^HANLON VX.XX ######
0005           remark ABOVE COPYRIGHT NOTICE MAY NOT BE ALTERED OR REMOVED
0006 MENU      mount screen SMENU
0007 START     N.2.0.1 = <0>
0008           enter N.2.0.1
0009           goto line on value of N.2.0.1 maximum gotos 10 if error goto START
0010 SENSIBLE  execute .Com file SENSIBLE
0011 ENTFLE    execute .Run file ENTFLE
0012 SENSCRN   execute .Com file SENSCRN
0013 SENSCMD   execute .Com file SENSCMD
0014 SENSINIT  execute .Com file SENSINIT
0015 SENSCOMP  execute .Com file SENSCOMP
0016 SENSRKEY  execute .Com file SENSRKEY
0017 SENSRSTC  execute .Com file SENSRSTC
0018 SENSGEN   execute .Com file SENSGEN
0019 SENSINQR  execute .Com file SENSINQR
```

ARRAY.SCC



```
                        +------------+
                        |   ARRAY    |
                        +------------+
This program will allow you to automatically create array element entries
(field definitions) in the Data Dictionary.  Enter the name of the first
array element, the number of elements that you wish to create, and the first
element number.  You can add elements to an existing array by indicating
which element to begin with.   This program will check for duplicate entries.
You must INITIALIZE or RESTRUCTURE the data file after adding the new array
elements.

+------------------------------------------------------------------------------+
|  Array Name: ************    Number of Elements: ###    Begin With: ***     |
|                                                                              |
|                                                                              |
|                                          File Name: ********                 |
|                                         Field Type: *                        |
|                                         Field Size: ###                      |
|                                           Decimals: #                        |
|                                                                              |
|  Field Name: ***************           Field File: ***********************  |
|                                                                              |
+------------------------------------------------------------------------------+
```

Field name	File	Size	Col	Row	Key
S.12.1	MEMORY	012	016	13	N
N.3.0.1	MEMORY	3	052	13	N
S.3.1	MEMORY	3	071	13	N
FLD.FILE	FLDFLE	8	052	15	N
FLD.TYPE	FLDFLE	1	052	16	N
FLD.SIZE	FLDFLE	3	052	17	N
FLD.DEC	FLDFLE	1	052	18	N
FLD.FLD.NAME	FLDFLE	15	016	20	N
FLD.FF.NAME	FLDFLE	23	052	20	Y

ARRAY.SRR

```
            remark THIS PROGRAM WILL AUTOMATICALLY CREATE ARRAY ELEMENTS (FIELD
            remark DEFINITIONS) IN THE DATA DICTIONARY--FIELD INFORMATION (FLDFLE).
            remark AFTER EXECUTING THIS PROGRAM, RETURN TO THE DATA DICTIONARY--FILE
            remark INFORMATION (RECFLE) AND ENTER THE NAME OF THE DATA FILE.
            mount screen ARRAY
            remark USER ENTER ARRAY NAME (12 CHAR.)
START       enter S.12.1
            remark USER ENTER # OF ELEMENTS IN ARRAY
            enter N.3.0.1
            remark USER ENTER BEGINNING ELEMENT #
            enter S.3.1 mask <###
            enter FLD.FILE
            enter FLD.TYPE
            enter FLD.SIZE
            enter FLD.DEC
            N.3.0.2 = (S.3.1)
            remark PAD WITH 0'S FOR 001, 002, 003, ETC..
            S.3.1 = fill leading chrs with 0
            remark ONLY SIGNIFICANT CHARACTERS IN FIELD WILL BE MOVED.
            S.12.1 = trim spaces trailing
            remark CREATE CONCATENATED KEY OF FILENAME PLUS FIELDNAME.
SAVE.GRP    FLD.FF.NAME = (FLD.FILE)+(S.12.1)+(S.3.1)
            remark VERIFY THAT FIELD DOES NOT ALREADY EXIST
            find exact rec using field FLD.FF.NAME  on error goto CONTINUE
            goto DUPLICATE
            remark CREATE FIELDNAME
CONTINUE    FLD.FLD.NAME = (S.12.1)+(S.3.1)
            save rec in file FLDFLE no confirm / no clear buffer
            remark CLEAR RECORD NUMBER SO THAT NEXT SAVE WILL APPEND TO END OF FILE.
            clear record number in file FLDFLE
            remark INCREMENT THE COUNTERS FOR NEXT ELEMENT AND TOTAL.
            N.3.0.2 = (N.3.0.2)+<1>
            N.3.0.3 = (N.3.0.3)+<1>
            remark TEST COUNTER.  ARE ALL ELEMENTS CREATED?
            if N.3.0.3 = (N.3.0.1) goto CLEAR
            remark MOVE COUNTER TO STRING FIELD FOR CONCATENATION.
            S.3.1 = (N.3.0.2)
            remark PAD LEADING CHARACTERS WITH ZEROS.
            S.3.1 = fill leading chrs with 0
            goto SAVE.GRP
```

```
0041            remark CLEAR SCREEN TO ENTER THE NEXT ARRAY.
0042 CLEAR      clear buffer in file MEMORY
0043            clear buffer in file FLDFLE
0044            remark IF DUPLICATE ARRAY ENTERED, PRINT MESSAGE AND CLEAR SCREEN.
0045 DUPLICATE  print at col 000  row 00 message THIS ARRAY ELEMENT ALREADY EXISTS.
                RE-ENTER NAME OR BEG. #
0046            clear buffer in file MEMORY
0047            clear buffer in file FLDFLE
0048            goto START
```

Introduction

The SENSIBLE SOLUTION Language is a highly advanced business applications programming language; it makes frequent access to disk drives and uses some of the more sophisticated capabilities of your display terminal. In order to perform properly the SENSIBLE SOLUTION must know exactly which "control codes" your display terminal and printer use for communication.

By following the instructions in this Installation Manual you will be able to embed or "install" this information in the SENSIBLE SOLUTION. Once you have done this, the input/output portion of SENSIBLE SOLUTION will run quickly and fault-free on your computer system.

"Installing" SENSIBLE SOLUTION on your computer hardware system is easy. Read the next four sections of this manual carefully and follow all of the instructions. The actual installation process -- running SENSETUP -- will only take you about ten minutes.

Get to work and you'll soon have The SENSIBLE SOLUTION Language "up and running" on your machine!

System Requirements

The following are the basic system requirements for the installation and operation of **SENSIBLE SOLUTION** software:

(1) **Operating System:** SENSIBLE SOLUTION requires a **CP/M, MP/M, MS-DOS** or similar, compatible operating system. Most hardware which supports such an operating system will run SENSIBLE SOLUTION. Examples of these operating systems are: CP/M, MP/M, MS-DOS, DPC/OS, TurboDOS, PC-DOS, MmmOST, n/STAR, CP/NET.

(2) **RAM Memory:** SENSIBLE SOLUTION requires RAM memory of 48k TPA or greater (free user area exclusive of operating system requirements):

(a) CP/M, MP/M	48k RAM TPA (DEC Computer requires 96k)
(b) CP/M-86	128k
(c) MS-DOS (PC-DOS)	128k (Victor Computer requires 256k)
(d) MP/M-86	128k

(3) **Mass Storage:** SENSIBLE SOLUTION requires mass storage capability of at least two floppy Disk Drives, each with at least **320k bytes** usable (floppy disk) storage capacity after formatting. Additional drives, disk capacity, or hard disks will increase system performance. Hard disks are recommended.

(4) **CRT Terminal:** SENSIBLE SOLUTION requires a CRT (Video) terminal of the following minimum requirements:

(a) ASCII serial type or ANSI compatible
(b) Screen Display: 24 (lines) by 80 (columns)
(c) Direct Cursor Addressing (absolute)
(d) Clear to End of Line
(e) Clear Screen

(5) **Printer:** SENSIBLE SOLUTION requires a printer with the following minimum requirements:
(a) ASCII type
(b) 80 column or more (e.g. 255 column compressed print)

Pre-Installation Instructions

1. Please fill out the User License Agreement and return it to your Dealer or O'Hanlon Computer Systems, Inc. Your signed Registration Card must be on file with O'Hanlon in order to preserve your warranty.

2. Make copies of your original SENSIBLE SOLUTION diskettes. Refer to your computer's operating system manual for the correct diskette copying procedure. Never use the original SENSIBLE SOLUTION diskettes as "working" diskettes.

3. The SENSIBLE SOLUTION Language as supplied is for a specific operating system (CP/M, CP/M-86, MP/M, MP/M-86, MS-DOS, TurboDOS, DPC/OS, etc) and is either for a single or multi-user operating system.

SENSIBLE SOLUTION File Extensions

filename.SCC	Screen format files
filename.SRR	Source code files
filename.RUN	Compiled SENSIBLE SOLUTION program files (pseudo code)
filename.MS	Master data file
filename.KS	Key file (for the .MS file)
filename.COM	Compiled executable programs (machine code)

Files Supplied With The SENSIBLE SOLUTION Language

(The Mail List Demonstration Module)

MAILLIST.RUN	SENSIBLE SOLUTION program
MAILLIST.KS	key file
MAILLIST.MS	data file
MAILLIST.SCC	screen format file
MAILLIST.SRR	source code file

(The SENSIBLE SOLUTION Language Main Menu Module)
FLOPMENU.RUN (for floppy disk systems)
MENU   .RUN (for hard disk systems)
MENU   .SCC
MENU   .SRR

(The System Error Message Module)
ERRFLE .KS
ERRFLE .MS

(The Data Dictionary Module)
FLDFLE .KS
FLDFLE .MS
RECFLE .KS
RECFLE .MS

(The Temporary Memory Variable Files)
MEMORY .MS
MEMORY .KS

(The System Definition Files)
SENSCTRL.MS
TERMDEFS.MS

(The System Configuration/Installation Program)
SENSETUP.COM

(Multi-User Utility)
SENSFREE.COM

(The SENSIBLE SOLUTION Language Main Menu Programs)

SENSIBLE.COM	language executive program
ENTFLE .RUN	data dictionary maintenance program
SENSCRN .COM	screen painting program
SENSCMD .COM	source code editor program
SENSINIT.COM	file initialization program
SENSCOMP.COM	compiler program
SENSRKEY.COM	key file re-key program
SENSRSTC.COM	data file restructure program
SENSGEN .COM	automatic program generator program
SENSINQR.COM	quick report generator program

Installation Instructions

Installing the SENSIBLE SOLUTION Language on your system involves two procedures: (1.) transferring the files to the proper drive locations on your system and setting the appropriate system attributes and (2.) running the terminal/printer/system installation program, SENSETUP.COM. If you are using a floppy disk system, you will also have to (3.) re-name the menu programs.

In the following instructions we will make frequent reference to certain file names that end with the extension "**.COM**". This particular file name extension is unique to the CP/M operating system. If you purchased the CP/M-86 version of The SENSIBLE SOLUTION, you should note that those files will be named with the extension "**.CMD**". Similarly, if you purchased the MS-DOS version of The SENSIBLE SOLUTION, those files will be named with the extension "**.EXE**".

Your computer system must be a single-user/floppy-disk or a single-user/hard-disk for CP/M, CP/M-86, or MS-DOS or it must be a multi-user/hard-disk for MP/M, MP/M-like, or MP/M-86. Each one of these three types of systems will require its own particular installation procedure. Use one of the 3 following sets of instructions that applies to your system.

I
SINGLE-USER/FLOPPY-DISKETTE INSTALLATION
CP/M and MS-DOS

1. If you have not already done so, make working copies of your SENSIBLE SOLUTION distribution diskettes. This can be done by making an image copy of the diskettes using the disk copy utility provided by your hardware manufacturer.

2. We assume that the floppy drives are designated 'A' and 'B', that drive 'A' is the default (logged on) drive, and that each drive has a minimum of 320K bytes of disk storage.

3. Create a bootable program diskette with the necessary operating system utilities (PIP, COPY, STAT, etc.). Test the boot (the computer's cold-start instructions) before doing anything else. See your computer's

operating system manual.

4. All of the necessary SENSIBLE SOLUTION Language files will be on two diskettes labeled "Disk 1" and "Disk 2". Copy (using PIP or COPY) all of the files from Disk 1 onto a diskette that will be used in drive 'A' and all of the files from Disk 2 onto a diskette that will be used in drive 'B'.

Disk 1

```
ENTFLE  .RUN
ERRFLE  .MS
ERRFLE  .KS
FLDFLE  .MS
FLDFLE  .KS
FLOPMENU.RUN
MAILLIST.RUN
MAILLIST.SRR
MAILLIST.SCC
MAILLIST.MS
MAILLIST.KS
MEMORY  .MS
MEMORY  .KS
MENU    .RUN
MENU    .SRR
MENU    .SCC
RECFLE  .MS
RECFLE  .KS
SENSCTRL.MS
SENSETUP.COM
SENSIBLE.COM
SENSINQR.COM
TERMDEFS.MS
SENSMAIN.EXE (used on MS-DOS systems)
```

Disk 2

```
ERRENT  .RUN
SENSCMD .COM
SENSCOMP.COM
SENSCRN .COM
SENSFREE.COM
SENSGEN .COM
SENSINIT.COM
SENSRKEY.COM
SENSRSTC.COM
```

5. Run the installation program SENSETUP.COM to configure your display terminal and printer and to set various system parameters. For details, see 'SENSETUP' in the following section of this Installation Manual.

6. Any time that SENSIBLE SOLUTION is run it immediately executes a program called "MENU.RUN" to bring up the main menu of the language. The proper menu program for a floppy-diskette installation such as this is called "FLOPMENU.RUN". This program is on drive 'A' and it must be re-named. Re-name the following two files with your operating system

re-name command, "REN":

```
Re-name MENU.RUN as HARDMENU.RUN
Re-name FLOPMENU.RUN as MENU.RUN
```

7. Type **SENSIBLE** followed by a carriage return and you're ready to begin using the SENSIBLE SOLUTION Language. At this stage a message will appear on your screen indicating that this is a demonstration version and has a 150 record data limit imposed on it. In order to remove the 150 record limit on you SENSIBLE SOLUTION software, telephone O'Hanlon Computer Systems and we will give you a special code that will remove the data limit from your language. This unlocking process is called "serialization."

   Before calling O'Hanlon Computer Systems to get your software serialized, make sure that you have the version of SENSIBLE SOLUTION that you want -- single-user or multi-user. When you call O'Hanlon, you will talk with a representative for serialization instructions. You must be at your computer terminal while you talk to the representative. The representative will ask you to give your name and address or your company's name and address and the number of operators that your system will be licensed for. After the serialization, this information will appear on your screen every time that SENSIBLE SOLUTION is run.

II
SINGLE-USER/HARD-DISK INSTALLATION
CP/M, CP/M-86, and MS-DOS

1. If you have not already done so, make working copies of your SENSIBLE SOLUTION distribution diskettes. This can be done by making an image copy of the diskettes using the disk copy utility provided by your hardware manufacturer.

2. Use the PIP or COPY utilitys to transfer all of the SENSIBLE SOLUTION files from Disk 1 and Disk 2 onto your system's hard disk.

3. Run the installation program SENSETUP.COM to configure your display terminal and printer and to set various system parameters. For

details, see 'SENSETUP' in the following section of this Installation Manual.

4. Type **SENSIBLE** followed by a carriage return and you're ready to begin using the SENSIBLE SOLUTION Language. At this stage a message will appear on your screen indicating that this is a demonstration version and has a 150 record data-limit imposed on it. In order to remove the 150 record limit on you SENSIBLE SOLUTION software, telephone O'Hanlon Computer Systems and we will give you a special code that will remove the data limit from your language. This unlocking process is called "serialization."

   Before calling O'Hanlon Computer Systems to get your software serialized, make sure that you have the version of SENSIBLE SOLUTION that you want -- single-user or multi-user. When you call O'Hanlon, you will talk with a representative for serialization instructions. You must be at your computer terminal while you talk to the representative. The representative will ask you to give your name and address or your company's name and address and the number of operators that your system will be licensed for. After the serialization, this information will appear on your screen every time that SENSIBLE SOLUTION is run.

III
MULTI-USER/HARD-DISK INSTALLATION
MP/M-type and DPC/OS-type

(TurboDOS, MmmOST, n/STAR, CP/NET, MP/M-86, etc.)

1. If you have not already done so, make working copies of your SENSIBLE SOLUTION distribution diskettes. This can be done by making an image copy of the diskettes using the disk copy utility provided by your hardware manufacturer.

2. Use the PIP utility to transfer all of the SENSIBLE SOLUTION files from Disk 1 and Disk 2 onto your system's hard disk. You will need to refer to your operating system manual for complete instructions on how your files must be set for multi-user activity. Some multi-user operating systems, for example, will require that you set all of the SENSIBLE SOLUTION main menu files (SENS*.COM, etc.) to "read/only" and "system".

3. Run the installation program SENSETUP.COM to configure your display

terminal and printer and to set various system parameters. For details, see 'SENSETUP' in the following section of this Installation Manual. The first menu option of SENSETUP, "Define system installation" will allow you specify whether or not your system is multi-user. Specify multi-user.

SENSCTRL.MS, which among other things contains your display terminal's communication codes, is created by the installation program SENSETUP.COM. If different types of display terminals are to be used on your system, you will need to move a unique copy of SENSCTRL.MS, for each type of terminal, onto each user area.

Additional multi-user considerations are discussed in the Reference Manual under the section "Data Structures".

4. Type **SENSIBLE** followed by a carriage return and you're ready to begin using the SENSIBLE SOLUTION Language. At this stage a message will appear on your screen indicating that this is a demonstration version and has a 150 record data-limit imposed on it. In order to remove the 150 record limit on you SENSIBLE SOLUTION software, telephone O'Hanlon Computer Systems and we will give you a special code that will remove the data limit from your language. This unlocking process is called "serialization."

   Before calling O'Hanlon Computer Systems to get your software serialized, make sure that you have the version of SENSIBLE SOLUTION that you want -- single-user or multi-user. When you call O'Hanlon, you will talk with a representative for serialization instructions. You must be at your computer terminal while you talk to the representative. The representative will ask you to give your name and address or your company's name and address and the number of operators that your system will be licensed for. After the serialization, this information will appear on your screen every time that SENSIBLE SOLUTION is run.

SENSETUP

The Display Terminal and Printer Installation Program

The SENSIBLE SOLUTION Language and the SENSIBLE SOLUTION Run-Time Module must "know" the exact input/output codes that your display terminal uses. In addition, SENSIBLE SOLUTION must know some of the parameters that your printer employs like the number of lines per printed page, the width of the page, etc.. The SENSETUP program is used to embed or "install" this required information in the run-time portion of SENSIBLE SOLUTION.

After you have finished loading all of the files that comprise the run-time portion of the language into your computer's disk storage, you will be ready to perform the installation procedure. From the system level type the file name SENSETUP:

```
A> SENSETUP [RETURN]
```

The following menu will appear on your display screen:

```
SENSIBLE SOLUTION(tm) Setup Program
===================================

     1) Define system installation
     2) Define printer
     3) Define terminal
     4) Save definitions
     [ESC] to abort without update

Your choice:
```

DEFINE THE SYSTEM INSTALLATION:

Choose selection number 1, "Define system installation," by pressing "1" on your terminal keyboard and the following list will appear on your screen:

```
System definitions for this installation --

*     "any character" template marker
A     "alphabetic" template marker
#     "numeric" template marker
?     "user-defined" template marker
.     "decimal point" character
S     Short (MM/DD/YY) or Long (MM/DD/YYYY) date format
U     US (MM/DD/YY) or European (DD/MM/YY) date style
@     Data Dictionary drive ("@" = currently-logged drive)
0     File locking -- 0=none, 1=MP/M, 2=DPC/OS
Y     Enable TRACER?
N     REQUIRE [ENTER] on all inputs?
      (otherwise, "enter and go" when field full)

==>   Press the "M" key to modify these, any other key to accept.
```

This portion of SENSETUP will allow you to either accept the current definitions of SENSIBLE SOLUTION or to change them as you wish. You may change the field window prompts (represented by the first four lines on this list), the location of the Data Dictionary, file locking and tracer definition, and the carriage return requirement.

When this list first appears on your screen, consider whether you want to modify any of the system definitions listed. If you want to change a definition, type the letter "M". The program will present the first line of the system definition list and the cursor will be positioned on the current definition. If you want to change it, simply type in the character you want to appear for that kind of prompt. Most users will be satisfied with the current definition displayed in the left column of the list. If you want to accept this definition, press the [RETURN] key and the cursor will advance to the next line on the list.

The SENSIBLE SOLUTION Language and nearly all programs written in the language will make frequent use of preformatted screen displays called "templates." As you run SENSIBLE SOLUTION programs these templates will be displayed on your screen. The templates are usually composed of verbiage, dotted lines or graphic character lines, and field windows. Every time the cursor appears in a blank field window to await an entry by you, a line of characters the exact width of the field will be displayed. This is called a "prompt." In addition to the length of the field, the prompt characters displayed in the field window will indicate the actual type of data that must be entered into the field window -- any character from the keyboard, alphabetical characters only, numeric characters only, or any user-defined character. The first four lines of this definition list will allow you to either accept the current definition of prompt characters or to redefine the prompt characters.

The fifth line on the list will allow you to specify the character that you want to use as a "decimal point". You will probably want to use either a "period" or the centered "dot" usually found on a numeric key pad.

The sixth and seventh lines on the list, will allow you to specify the type of date format you want to use. First specify the long or short format -- a two digit year versus a four digit year -- and then specify the common U.S. format or the European date style.

The eighth line on the list will allow you to specify the disk drive location of RECFLE.MS/.KS. RECFLE is part of the Data Dictionary, which maintains the definitions of all files and fields used by the system. One function of RECFLE is to maintain the actual disk drive location of all data files used by the system. Thus, you specify the drive location of RECFLE and RECFLE specifies the location of the data files. Accepting the current definition on this list, the "@" sign, will simply instruct SENSIBLE SOLUTION to look for RECFLE on the currently logged-on drive.

You are given the option of specifying a new drive location for a specific reason. The files that comprise the Data Dictionary, RECFLE and FLDFLE, can become very large. If this is the case with your system, you may need to conserve space by placing the Data Dictionary on a different disk drive. Use this list option to specify that drive location.

The next item on the list will allow you to specify whether or not you want to use file locking on your system. If you have a single-user system, answer with a "0". If your system is multi-user you will have to consult your system's operating system manual to determine the type of file locking system your computer uses. TurboDOS, CP/NET, and n-Star all emulate the MP/M operating system. MmmOST emulates DPC/OS.

The SENSIBLE SOLUTION has a debugging mode of operation that will allow you to execute programs one line at a time. If you want this capability on your system, accept the current value for the question: "enable TRACER?".

The last item on the list will allow you to specify whether or not the operator will be required to press the [RETURN] key after filling a field window with data. For example, if you are entering data into a 5 character long field window such as Zip Code and you have specified "N" on the list, the program will continue execution immediately after the fifth character is entered. Under this condition you would not be required to use [RETURN] after typing in the fifth character.

Remember, all you have to do to accept these standard definitions as shown on this list is to press any key on your keyboard except the "M" key. When you have finished with this system definition routine the Setup Program Menu will reappear on your screen.

DEFINE THE PRINTER:

Choose selection number "2", "Define printer," and the following menu will appear on your screen:

```
Printer definitions for this installation--

1)   Page size
2)   List Control sequences
     Any other key exits printer definition

Your choice:
```

Begin by selecting option "1", "Page size," and either accept the currently set parameters (66 max. lines per page, 60 printed lines per page, 132 columns per page) or specify new ones that will reflect your printer's specifications. When you are finished with this procedure the Printer Definition Menu will appear again.

There is a command in The SENSIBLE SOLUTION Language called "Print Control chrs." With this command you can use the full capabilities of your printer and video display by instructing them to print in bold face type, reverse video, double width print, special graphic character mode, etc.. "Print Control chrs" can recognize any one of 32 different predefined control sequences and then relay this sequence to the printer or terminal. Option "2" of the Printer Definition Menu will allow you to define these control sequences. If you wish to utilize this capability of SENSIBLE SOLUTION, select option "2" and follow the instructions displayed on your screen. You should remember, though, that you may redefine the List Control Sequences , which involves re-running SENSETUP, any time that you wish. If you are a first time user of the SENSIBLE SOLUTION, you may wish to skip option "2" for now and define control sequences as the need arises.

DEFINE THE TERMINAL:

The third option on the Setup Program Menu is "Define terminal." This procedure will allow you to embed all of your display terminal's input/output control codes within the run-time portion of SENSIBLE SOLUTION.

The terminal installation procedure varies slightly depending upon the operating system that you are using. If you are installing an MS-DOS version of SENSIBLE SOLUTION, the SENSETUP program will display an additional screen at the start of the terminal setup routine. If you are using CP/M, CP/M-86, or a multi-user version of SENSIBLE SOLUTION, this additional screen, which is discussed below, will not be displayed.

The MS-DOS Version Of SENSETUP:

Select option "3" and the following message will be displayed on your screen:

```
I/O style definition menu

Current I/O style in use is A)

     A) Terminal style
     B) IBM PC style
     C) TI PC style

Select style by letter, or [ENTER] to keep current style.
```

The terminal installation procedure for the IBM PC computer or a TI PC (Texas Instruments) computer is very simple. Select either option "B" or "C", whichever is appropriate, and press the [RETURN] key. You will immediately be returned to the Setup Program Menu. Next, select option 4 from that menu to save the definitions that you have specified in options 1, 2, and 3. After you have saved the IBM or TI definitions, you will be completely finished with the installation of SENSIBLE SOLUTION.

If you selected option "A" from the MS-DOS procedure described above or you are using an operating system other than MS-DOS (such as CP/M, CP/M-86, MP/M, etc.), select option "3" from the Setup Program Menu and proceed with following terminal installation procedure. From this point on, the terminal setup procedure is the same for all operating systems:

```
Terminal definition menu

Current terminal in use is TERMINALNAME

-- use different terminal? (Y or N)
```

If no terminal name is shown on your screen or if you want to specify a new terminal definition for SENSIBLE SOLUTION, answer "Y" to the question. A list of predefined terminal setups will be displayed on your screen:

```
A) TeleVideo 925
B) TeleVideo 950
C) Hazeltine Esprit III
D) KayPro 10
   etc.

Select terminal by letter, [ENTER] to continue, [ESC] to restart list
or enter "0" to define a new terminal
```

If your particular terminal is shown on the list, all you have to do to install it is type the appropriate letter. All of the definitions displayed on your screen are stored in a file named TERMDEF.MS. After you indicate the desired terminal definition, the Terminal Modification Menu will be displayed:

```
Terminal Modification Menu

Current terminal is <terminalname>

A) Terminal Name                      K) Graphics Mode ON
B) Screen size                        L) Graphics Mode OFF
C) Cursor Positioning                 M) Cursor ON
D) Clear Screen                       N) Cursor OFF
E) Clear to End of Screen             O) Insert screen line
F) Clear to End of Line               P) Delete screen line
G) Background                         Q) Box-drawing characters
H) Foreground                         R) Field-editing characters
I) Reverse Video                      S) TEST TERMINAL DEFINITION
J) Normal Video                    [ESC] to exit terminal modification
```

Choose option "S", "TEST TERMINAL DEFINITION," and proceed through the instructions displayed on your screen. SENSETUP will test your current terminal setup. If the test procedure does not yield the correct results, you may have either specified the wrong terminal from the list or the exact terminal you are using does not match the list. If you do pass the test, however, you are nearly finished with the installation process. Simply type in option "4" from the Setup Program Menu, "Save definitions." A message will be displayed on your screen asking if you want to save this current terminal definition -- answer "Y". You are now completely finished with the installation procedure.

If you are not among the norm and your terminal is not listed, you will have to do some extra work. First try selecting option "3" again, "Define terminal," and make sure you selected the correct terminal on the list. If your terminal or a similar terminal is not listed, you will have to go through the following procedure to configure your terminal.

Answer "Y" to the following screen message:

```
     Terminal definition menu

     Current terminal in use is TERMINALNAME

     -- use different terminal? (Y or N)
```

The list of predefined terminals will appear and you should then answer the following prompt with a "0":

```
Select terminal by letter, [ENTER] to continue, [ESC] to restart list
or enter "0" to define a new terminal
```

After the following prompt, enter the name of the new terminal you wish to define and press [ENTER]:

```
Terminal name:
```

You will have to specify the screen size of your terminal and the cursor positioning hierarchy (row/column or column/row). Next, SENSETUP will ask you a variety of questions about the control codes that your terminal uses for communication. It will be up to you to glean the correct control codes from your terminal's factory reference manual, and then, using that information, answer the following questions as they are displayed on your screen. If you are a computer novice, we strongly recommend that you have your authorized SENSIBLE SOLUTION dealer, or someone else experienced with computer hardware, perform this procedure.

Once you have secured the correct control codes, the following procedure is really very ease to perform. SENSETUP will ask you a variety of questions from "row lead-in sequence" to "cursor positioning." Answer each question by pressing the correct key or series of keys on your terminal. For example, if a particular control code is "decimal 023," you would answer the question by typing in the control sequence "[CTRL] [W]." If you look on the ASCII code chart in the back of this manual, you will see that:

^W = decimal 023 = hexidecimal 017 = octal 027 = binary 00010111

The reference manual for your terminal may well specify the control codes in any one of these number types. You will have to use the ASCII table to make the conversion to the correct series of key strokes.

Before you get started, you should note that from now on there are no default values. If you are re-defining a terminal, you will see the previously specified key strokes displayed before each question. To accept those again, you must re-enter them. Pressing the [RETURN] key will not automatically accept them by default.

SENSETUP will now ask the following questions, answer with the correct series of key strokes:

ROW lead-in sequence

COLUMN lead-in sequence

Cursor Position TAIL (may be null)

Clear screen to blanks

Clear from cursor to end of screen

Clear from cursor to end of LINE

Set "background" (low-intensity)

Set "foreground" (high-intensity)

Set reverse video (dark characters on bright background)

Set normal video (bright characters on dark background)

Enable graphics mode

Disable graphics (normal text mode)

Turn cursor ON

Turn cursor OFF

Insert a blank line at cursor position

Delete the line the cursor is in (lines below move up)

Box-drawing characters
UPPER Horizontal line
LOWER Horizontal line
LEFT Vertical line
RIGHT Vertical line
Upper-left corner
Lower-left corner
Upper-right corner

Lower-right corner

Field-editing characters

Field-escape key (usually [ESC])
Left-arrow
Right-arrow
Up-arrow
Down-arrow
Insert character (usually (CTRL-I])
Delete character (usually (CTRL-D])
Clear field (usually [CTRL-U])

Some terminals are capable of producing simple graphics characters such as horizontal lines, vertical lines, and corners. These graphics characters can be used to print box shapes and bar graphs on your screen. When SENSETUP asks you to define "Box-drawing characters," you should specify the correct series of codes to produce each of these eight characters. If your terminal does not utilize any graphics characters, we suggest that you specify the following standard characters in order to produce boxes and lines:

SENSETUP Prompt	Specify
UPPER Horizontal line	"-"
LOWER Horizontal line	"-"
LEFT Vertical line	"|" or "!"
RIGHT Vertical line	"|" or "!"
Upper-left corner	"+"
Lower-left corner	"+"
Upper-right corner	"+"
Lower-right corner	"+"

When you have completed answering the last question, the Terminal Modification Menu will reappear on your screen. Type "S", "TEST TERMINAL DEFINITION," and perform the screen display test. If you fail part of the test, try selecting the problem area, re-configure that area, and try the test again. Remember, a little patience and a few experiments will probably yield a good terminal setup.

After your terminal passes the screen display test, type [ESC] (escape) and you will return to the Setup Program Menu. If you are completely satisfied with the current definition, select option "4" to save the terminal definition in a file named SENSCTRL.MS. In the future, every time you load SENSIBLE SOLUTION, the run-time portion will read your terminal's definition from the SENSCTRL.MS file and then begin execution.

ASCII Character Code Conversion Table

CHAR	DEC	HEX	OCT	BINARY	CHAR	DEC	HEX	OCT	BINARY
^@ NUL	000	00	000	00000000	@	064	40	100	01000000
^A SOH	001	01	001	00000001	A	065	41	101	01000001
^B STX	002	02	002	00000010	B	066	42	102	01000010
^C ETX	003	03	003	00000011	C	067	43	103	01000011
^D EOT	004	04	004	00000100	D	068	44	104	01000100
^E ENQ	005	05	005	00000101	E	069	45	105	01000101
^F ACK	006	06	006	00000110	F	070	46	106	01000110
^G BEL	007	07	007	00000111	G	071	47	107	01000111
^H BS	008	08	010	00001000	H	072	48	110	01001000
^I HT	009	09	011	00001001	I	073	49	111	01001001
^J LF	010	0A	012	00001010	J	074	4A	112	01001010
^K VT	011	0B	013	00001011	K	075	4B	113	01001011
^L FF	012	0C	014	00001100	L	076	4C	114	01001100
^M CR	013	0D	015	00001101	M	077	4D	115	01001101
^N SO	014	0E	016	00001110	N	078	4E	116	01001110
^O SI	015	0F	017	00001111	O	079	4F	117	01001111
^P DLE	016	10	020	00010000	P	080	50	120	01010000
^Q DC1	017	11	021	00010001	Q	081	51	121	01010001
^R DC2	018	12	022	00010010	R	082	52	122	01010010
^S DC3	019	13	023	00010011	S	083	53	123	01010011
^T DC4	020	14	024	00010100	T	084	54	124	01010100
^U NAK	021	15	025	00010101	U	085	55	125	01010101
^V SYN	022	16	026	00010110	V	086	56	126	01010110
^W ETB	023	17	027	00010111	W	087	57	127	01010111
^X CAN	024	18	030	00011000	X	088	58	130	01011000
^Y EM	025	19	031	00011001	Y	089	59	131	01011001
^Z SUB	026	1A	032	00011010	Z	090	5A	132	01011010
^[ESC	027	1B	033	00011011	[	091	5B	133	01011011
^\ FS	028	1C	034	00011100	\	092	5C	134	01011100
^] GS	029	1D	035	00011101	]	093	5D	135	01011101
^^ RS	030	1E	036	00011110	^	094	5E	136	01011110
^_ US	031	1F	037	00011111	_	095	5F	137	01011111

CHAR	DEC	HEX	OCT	BINARY	CHAR	DEC	HEX	OCT	BINARY
SPACE	032	20	040	00100000	`	096	60	140	01100000
!	033	21	041	00100001	a	097	61	141	01100001
"	034	22	042	00100010	b	098	62	142	01100010
#	035	23	043	00100011	c	099	63	143	01100011
$	036	24	044	00100100	d	100	64	144	01100100
%	037	25	045	00100101	e	101	65	145	01100101
&	038	26	046	00100110	f	102	66	146	01100110
'	039	27	047	00100111	g	103	67	147	01100111
(	040	28	050	00101000	h	104	68	150	01101000
)	041	29	051	00101001	i	105	69	151	01101001
*	042	2A	052	00101010	j	106	6A	152	01101010
+	043	2B	053	00101011	k	107	6B	153	01101011
,	044	2C	054	00101100	l	108	6C	154	01101100
-	045	2D	055	00101101	m	109	6D	155	01101101
/	047	2F	057	00101111	o	111	6F	157	01101111
0	048	30	060	00110000	p	112	70	160	01110000
1	049	31	061	00110001	q	113	71	161	01110001
2	050	32	062	00110010	r	114	72	162	01110010
3	051	33	063	00110011	s	115	73	163	01110011
4	052	34	064	00110100	t	116	74	164	01110100
5	053	35	065	00110101	u	117	75	165	01110101
6	054	36	066	00110110	v	118	76	166	01110110
7	055	37	067	00110111	w	119	77	167	01110111
8	056	38	070	00111000	x	120	78	170	01111000
9	057	39	071	00111001	y	121	79	171	01111001
:	058	3A	072	00111010	z	122	7A	172	01111010
;	059	3B	073	00111011	{	123	7B	173	01111011
<	060	3C	074	00111100	|	124	7C	174	01111100
=	061	3D	075	00111101	}	125	7D	175	01111101
>	062	3E	076	00111110	~	126	7E	176	01111110
?	063	3F	077	00111111	DEL	127	7F	177	01111111