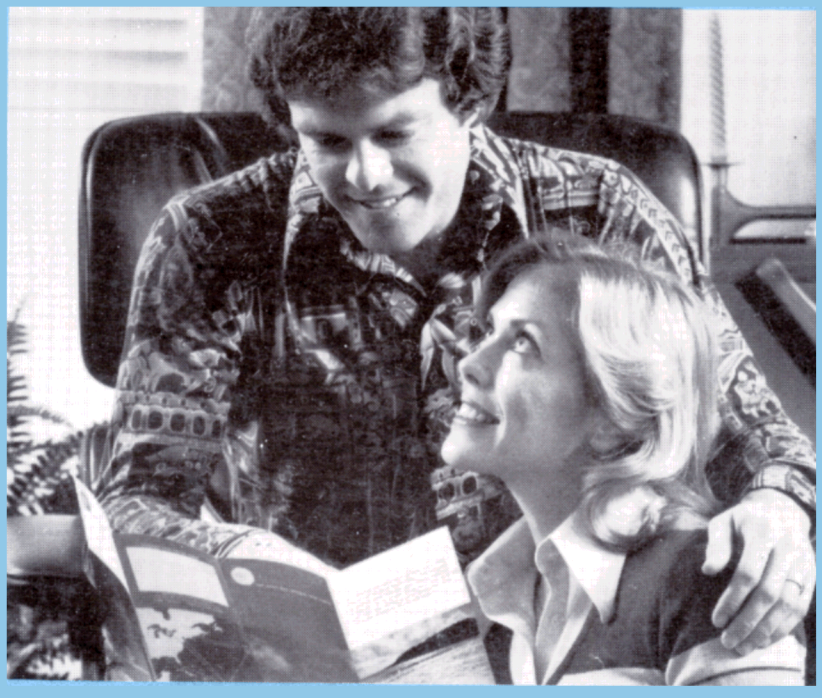
# PIMS

## Personal Information Management System



**Improve your life style**

**Learn how you can unleash the power of a personal computer for your own benefit. A ready-to-use data base management program.**

SCELBI Publications

# PIMS

## Personal Information Management System

SCELBI Publications
P. O. Box 3133, Milford, Connecticut 06460

IMPORTANT NOTICE



The information in this manual has been carefully reviewed and is believed to be entirely reliable. However, no responsibility is assumed for inaccuracies or for the success or failure of various applications to which the information contained herein might be applied.

# Foreword

You are about to unleash the power of a personal computer. You will discover untapped capabilities. To save money. To plan better. To locate important facts quickly. To eliminate the drudgery of routine chores. These all add up to a better life style for you! A new way of living which can be more pleasant, bring you more happiness and success.

Can this booklet really help? You bet it can. Until now only the programmers and designers could make such profitable use of a personal computer. *Personal Information Management System* has changed that. You don't have to be a programmer to use PIMS. It is an alternative to programming from your standpoint. All you do is define the job you want the microcomputer to do. Express yourself in simple commands and statements. Then the microcomputer plus PIMS does the rest.

In modern society, information has become a key to happiness and success. Your command of information is what gives you the power to succeed. You've heard of the information explosion? Now it's time for an explosion in your personal capacity to deal with a wealth of information. PIMS will help you in this exciting endeavor. You'll discover how you can find your new life style – with PIMS.

# Contents

Chapter 1

# A Microcomputer in Your Life

The microcomputer of today is a powerful tool. It offers the potential advantages of having a variety of skilled specialists serve you in the privacy of your own home or business. One minute it can perform accounting tasks. The next minute it can be serving as your appointment secretary. It is this versatility that makes the microcomputer such a useful tool for increased personal productivity.

How many of the following tasks do you have to perform on a regular basis? How many do you skip doing because it is "too much bother"?

1) Balance your checkbook.

2) Maintain a list of current department store charges.

3) Maintain a list of household valuables for inventory purposes as well as for possible use if it is necessary to file any insurance claims.

4) Keep a list of tax deductible expenses classified by various categories so that they can be readily tabulated when preparing the annual tax return.

5) Mail cards, invitations, or notices to friends, members of a club, business associates, etc.

*Accomplish More in Less Time*

Tasks such as these take a significant portion of any person's time. A better organized individual is a more efficient individual. If you had a convenient way of doing those tasks rapidly you could get more things accomplished in your life. You would free up time to do things you really liked to do – such as being creative with a pleasing hobby, or enjoying sports, or giving more time to your family.

Most creative people need the right circumstances and the tools to be creative. Possession today of tools which will be commonplace tomorrow can provide a definite advantage. Consider the possibilities that a personal microcomputer has to offer. It can readily be integrated into one's daily life to give you that advantage.

PIMS is for the computer novice. With the right equipment, this book is expected to serve as the "initiation rite" into the world of personal computer power. The program is complete and ready to use on computers like the Radio Shack TRS-80, level II. Various other machines may require some simple PIMS syntax modification if they are not compatible with Microsoft (a trademark of Microsoft, Inc.) BASIC language. Consult your computer dealer for assistance if needed. Once PIMS is properly installed in your machine, you should not need any help whatsoever to use PIMS effectively. Just follow the instructions and examples provided in this publication.

*". . . ready to use on computers like the Radio Shack TRS-80, level II . . ."*

Those interested in programming are encouraged to look at the program listing as often as possible. Quite soon you will start to identify patterns in the listing that correlate with what you see on your terminal. If you have read through an introductory BASIC language text, you should be able to understand the general flow of PIMS. You would also be in a position to customize it if desired.

**Equipment Required**

PIMS was written in Microsoft BASIC. It is thus designed to run on many popular microcomputer systems such as those manufactured by Apple, Cromemco, Imsai, Commodore, Radio Shack and other corporations serving the small computer market. The version shown in this publication has been thoroughly exercised on a Radio Shack TRS-80 level II system with 16K of user memory. Slightly modified versions of the program (having the input/output routines altered to suit the systems) have also been tested on a Commodore PET 2001 system as well as a MITS ALTAIR 8800b unit.

In order to execute PIMS you should have the following:

1) A microcomputer system equipped with a Microsoft or Microsoft compatible BASIC language interpreter. Many popular small computer systems have a BASIC interpreter residing in ROM (Read Only Memory). This method frees more RAM (Random Access Memory – also popularly referred to as "read and write" memory) for the user's programs.

2) Preferably at least 16K of user read and write memory. the more the better. The PIMS source listing will use about 8K of this memory. The remaining memory is then used to store the user's data base that PIMS will be able to manipulate. The more memory you have available in your system, the larger a data base you will be able to store and analyze in your computer! (Note: It is possible to demonstrate PIMS in as little as 8K of memory in some

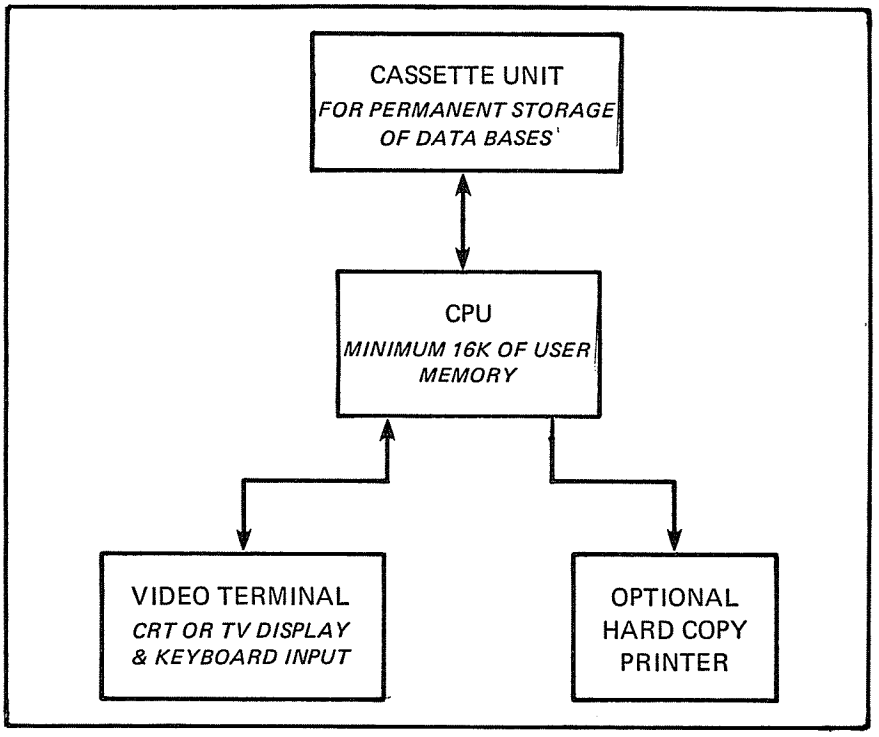
systems. However, the size of the data base in such cases is so restricted that the program would be of little practical use.)

3) A CRT (cathode ray tube) terminal for inputting information to the program via a keyboard and viewing data outputted by PIMS. The use of a video terminal greatly facilitates using PIMS in most situations. However, it is possible to utilize PIMS with just a hard copy terminal.

4) Optionally, a hard copy device such as a teletypewriter or other terminal having keyboard input capability coupled with an electro-mechanical display mechanism. Hard copy output capability is desirable even if you have a video display. There are situations, such as when preparing mailing labels, that only a hard copy device can handle appropriately.

5) A cassette tape unit so that you can store your data base permanently, then recall it when you want to continue working with the information. (By making minor changes to PIMS you can also use other types of external storage devices to save your data. Floppy disk units are becoming very popular these days and make an ideal accessory for use with PIMS.)

The accompanying diagram summarizes the system components recommended for use with PIMS.



## Chapter 2

# Jobs a Personal Computer Can Do for You

"What can really be done with a personal computer?" A lot of people have been asking this question. Until now, the unfortunate truth has been "probably not very much." The programs that would allow you to do meaningful things were not yet available. PIMS provides the kind of essential capability that business computer users have been capitalizing upon for years! It is one of the first such programs to be made available to the general public.

The key ingredient in the effective application of a computer is the ability to organize and manipulate information, information that is important to you personally. Manipulation of that information allows extraction of details that are meaningful and valuable to you personally. From a practical point of view, why do businesses spend millions of dollars on computer systems? They spend millions of dollars on these *tools* because they know they can make money by utilizing them! As an individual you can use the exact same principles in your personal life and derive great benefits. Benefits of the same kind and nature as those that have been accruing to businesses for the past thirty years.

### Practical Benefits

Such as? The saving of countless hours of your *time*. There is a saying in business that "time is money." Well, in personal matters, maybe it should be changed to "time is pleasure." Which would you rather be doing – balancing your check book or playing golf/tennis/bowling? Which would you rather be doing – sending out notices to all members of the bowling league (after calculating their individual bowling averages), or taking in a good movie with an attractive friend? Which would you rather have – the subjective opinion of your neighbor about the worthiness of a stock, or some good hard data about the performance of a company compared to other

firms in the same field?

The point is, you can choose pleasure over drudgery – if you have a computer capable of running a program like PIMS. The benefits of using a computer and such a program start with a savings in time. They are enhanced by the quality (and amount) of information that can be extracted and processed. This can lead to personal enrichment – by providing time for you to do other things and/or by giving you the ability to make better (more productive) judgements.

**Now You're Talking Business**

Of course, the word "personal" could apply to personal business matters. In that case the applications and benefits could be exactly like those associated with large computer systems. The only differences are those of scale. You can use PIMS to perform many business functions – from maintaining mailing lists, to keeping track of inventory, balancing the business checking account, and following business investments. Of course, we do not propose that PIMS can serve in the manner of a larger computer system. It is not capable of handling a large data base or working with the speed required to handle such a data base. But, it can do many of the kinds of tasks required in a small one or two man business, of which there are a great many in this world.

In a small business environment PIMS can help the entrepreneur extract and compile data in order to make "smarter" business decisions. PIMS can also reduce much of the drudgery associated with maintaining commonly used data bases. How much time do you spend using purely manual pencil and paper methods to keep your inventory and equipment maintenance records up to date? Have you ever wanted to find out how many widgets you had of a particular type – then abandoned finding the answer because it would mean scanning two hundred index cards, which you were in no mood to do. Perhaps you lost a sale because of that. Did you tell a telephone inquirer you "weren't sure if you had any of those left"? After you hung up the caller probably figured if you weren't sure, they surely were not going to drive down to visit your place of business!

**Let's Get Specific**

No matter what task you have in mind for your computer, you need to start with a *data base*. What's that? Your data base is all the relevant information you will be using for a particular application. Take names and addresses for example. You probably keep a variety of

records of names and addresses: friends, relatives, stores you do business with, neighbors, bowling league members, etc. Collectively they constitute your personal data base of names and addresses. You use this information when you wish to write a letter, pay a bill, or send out holiday greeting cards.

Chances are that you keep a separate file for each category of address or type of task. The total number of records can be quite large. Now, the separate files and the large number of records may not pose a problem for everyday applications. But, to illustrate a point, let's suppose you are going to do something special.

**You're Going to Throw a Party**

You're planning a really big party. You want to invite everyone you know within a 25-mile radius. Using conventional methods, you would have quite a job ahead of you in putting the mailing list together. Some of the addresses are in your Christmas card list. Some are on the company's employee list. Others are in your telephone directory. You will need to scan through all those sources and copy the names and addresses of invitees living near your house. That's sure a time-consuming task. Not much fun, either.

**Let the Computer Do It**

How would this chore be any different if you were to use a personal computer? What if your data base of names and addresses had been recorded in your computer's memory? Then, using PIMS, the computer could compile the list for you. First you could sort your entire data base according to zip code order. Sound like a week's work? Not for a computer! Once in zip code order, you could select those addresses in your locale based on zip code, using only those within a certain range. Now you have your list! Suppose you wanted to avoid sending invitations to utility companies and stores that appeared in your data base? No problem. Just ask the computer to sort out those categories of addresses. You will be amazed at the computer's speed in performing tasks such as these.

**Checking Up on Your Checkbook**

Another data base worth keeping in your computer is the information contained in your checkbook. This is particularly useful if you pay for most of your expenses by check. To start with, the computer will save you time at month's end when it is necessary to balance the check book. But that is just the beginning. Maybe you have a grocery budget and want to know how much you have paid the supermarket lately. Ask the computer. It can selectively sum all the checks made payable to that particular store. You will have an answer in seconds.

How about automobile repairs? Want to know what you have spent in that category during the last six months? A convenient sort by category of expense will give you the answer. And when it comes time to prepare your income tax forms, you can get a sum for medical expenses, exempt contributions or any other significant expense category.

**Is It Like Radar?** An airport uses radar to keep track of the aircraft for which it is responsible. You may or may not have your own airplane. But you probably do have a number of possessions that you need to keep track of. This information can be useful for insurance purposes. You will have an ever-ready record of valuables. Your inventory of possessions could also include warranty information and indicate where each item was purchased. You can record where you have stored a particular item and make note of what you have loaned to friends.

When it comes to hobbies, the inventory feature can really perform. If you are a coin or stamp collector, you can catalog your entire collection in a way that allows much interesting analysis. Summing the value of your collection at any time is an easy trick. Want to know which are your most valuable items? Sort your collection by order of value. This power can be beneficial when making decisions regarding the buying and selling of items.

As another example, let's use the workshop of an electronics hobbyist. A computerized inventory of components can be quite a boon when planning a construction project. Not only can you find out whether or not you have a particular component, but you can quickly identify the container or bin in which it is stored. You can even call for the name and address of the supplier for those parts that are not in your inventory. All these things can be done with speed and convenience. The only prerequisite is that you keep all the vital data stored as part of your computer data base.

**Hamming It Up** Amateur radio operators maintain logs of their transmitting activity. Most operators today record the information in a more or less permanent notebook. In addition to recording technical information, such as the type of transmission mode being utilized and the amount of power emitted, operators typically like to record other information such as: the date/time of a contact with another amateur radio station, the identifying call of the station they work (converse with), the other station's location, the frequency band on

which the communication took place, and signal or "RST" reports exchanged. An amateur might also want to keep track of data such as the type of antenna being used during a particular contact or keep notes of personal interest such as the other operator's hobbies, nickname, equipment used, and so forth.

All of this kind of information could be maintained in a computer file using PIMS. Of course, it could also be recorded in a paper notebook or on index files. However, the computer method offers all sorts of advantages when it comes to using the information.

*An Intelligent Log Book*

For instance, every few months a person could ask the computer to arrange the radio contacts by station call signs. This single operation by PIMS would group the information so that it yielded interesting data that might not otherwise ever become apparent. It would reveal the number of times the same station had been contacted. It would show the different sections of a country or the different countries in the world that had been contacted.

The computer could group the contacts made with each type of antenna used at the station. Comparisons could then be made to determine which antenna was providing the best signal reports – both on the receiving and transmitting sides. Conversely, one could arrange the data by RST signal report, then compare reports to see if a particular antenna – or even the time of day – had any effect on the average signal levels reported! This type of valuable operating information, achievable in just a few minutes using PIMS, might take hours to obtain by manual methods.

*QSL, OM?*

Amateur radio operators frequently exchange post cards called "QSL" cards after they have made radio contact to verify the communication. These verifications are frequently coveted as proof of radio contact. Many organizations offer certificates and other prizes to radio operators who achieve certain goals – such as contacting radio stations in 100 or more countries. (That is no easy feat! It takes considerable time and skill for an amateur radio operator to achieve such a record.)

Now the collecting of QSL cards can be a project in itself. Some radio operators do not like to bother with the fuss and expense of exchanging the cards. They sometimes have to be coaxed into providing written verification of a radio contact. Cards also sometimes get lost in the mail, especially if they are going to or coming from foreign lands. Obtaining proof of radio contact via a QSL card can often prove as elusive as the effort to locate and contact the desired signal on the air. Can PIMS help here?

Yes! PIMS can be used to keep track of the status of a QSL card exchange. The name and address of a contact may be stored. Notes can be made as to whether a card has been sent and/or received. A mailing list can be prepared in minutes. You can elect to have labels printed just for those addressees from whom you have not yet received a card. In seconds you can review your QSL status. Such capability is indeed appreciated by those who chase the sometimes elusive QSL cards.

**We Mean Business, Too**

How much money do your customers owe you? This is important information to the small businessman. Getting a clear picture of receivables can be difficult and time consuming without a computer. PIMS will allow you to computerize your receivables data base. Then chores like summing outstanding amounts can be done quickly and easily. You can sort out those accounts which are badly overdue, do a quick check on an individual's account for credit purposes, or prepare a monthly statement of an account.

The analytical power of PIMS combined with a personal com puter lends well to sales analysis. In business, you need to know which salesmen are performing best, or which product is making the greatest contribution to sales volume. This kind of knowledge enables you to focus on opportunities for expansion, identify problems to be solved, and to streamline the administration of your business.

You can conduct these kinds of analyses once your sales information has been assembled into a computer data base. It then becomes a simple matter to sum the weekly or monthly sales of each salesman. You can generate a report showing the performance of your salesmen with each of several product lines. And, there is a simple procedure for determining which salesman did the highest sales volume with a particular product.

## Chapter 3

# What Is Information Management?

We shall start this chapter by defining the key terms that we shall be using when talking about information management using PIMS.

CHARACTERS – are the smallest units of information that we can deal with using PIMS. A character may be a letter of the alphabet (A, B, C . . .), a numerical digit (1, 2, 3 . . . ), symbol or punctuation mark (+, $, * . . . ), or control code ("line enter" or "carriage return"). It is the unit that is received by the computer when a single key is depressed on the operator's keyboard. It is the smallest amount of information that can be sent to the display unit by the computer. A character is expected to be encoded when being transmitted to and from the computer in ASCII format.

STRINGS – consist of any group of characters that are manipulated as an entity.

WORDS – are groups of characters that do not contain any spaces.

FIELDS – are composed of characters, words or strings. A field in PIMS is always terminated by a carriage return or by depressing the "line enter" key. Fields usually, but not necessarily, contain information of a specific nature and are often referenced by their logical function. Thus, a field may be designated as a "name field" because it is used to hold a person's name. Note that the name may consist of several words, such as the first and last name. A line of data containing a street address might be referenced as an "address field."

NUMERIC FIELDS – in PIMS can only contain the numeral digits 0 through 9 and a few special characters such as the decimal point (.) or minus (–) sign. A field is designated as numeric at the discretion of the operator at the time that a data base is organized as will be explained later in this manual. Numeric fields in PIMS have a special attribute. The contents of selected fields can be automatically summed to yield column totals.

ALPHANUMERIC FIELDS – in PIMS may contain any valid character, whether a letter of the alphabet, number, symbol or punctuation mark (with special consideration when commas are used as will be noted later).

RECORDS – are composed of from one to a maximum of ten fields. The number of fields in all the records of a data file are determined by the user of PIMS for the particular application. A record usually forms a logically complete structure in terms of the information it contains. Thus, a record in a mailing list application might hold a person's name and address, category within the list and telephone number.

DATA – is the information that you store in records when using PIMS. It is the technical name for the "knowledge" that your computer will be manipulating for you!

DATA FILES – are composed of records. All the records for a given application that can reside in the computer at one time make up a data file. The contents of a data file can be stored on an external mass storage device such as a magnetic tape recorder. In practice, a data file is usually established for each type of application. A list of names and addresses might make up the data file for a mailing list application. A list of individual components or parts, the quantity purchased, and the price at which they were purchased might make up the data file for an inventory application.

DATA BASES – may be viewed as all the information stored in data file(s) that pertain to any or all applications. The total amount of information that you have to manipulate is your data base.

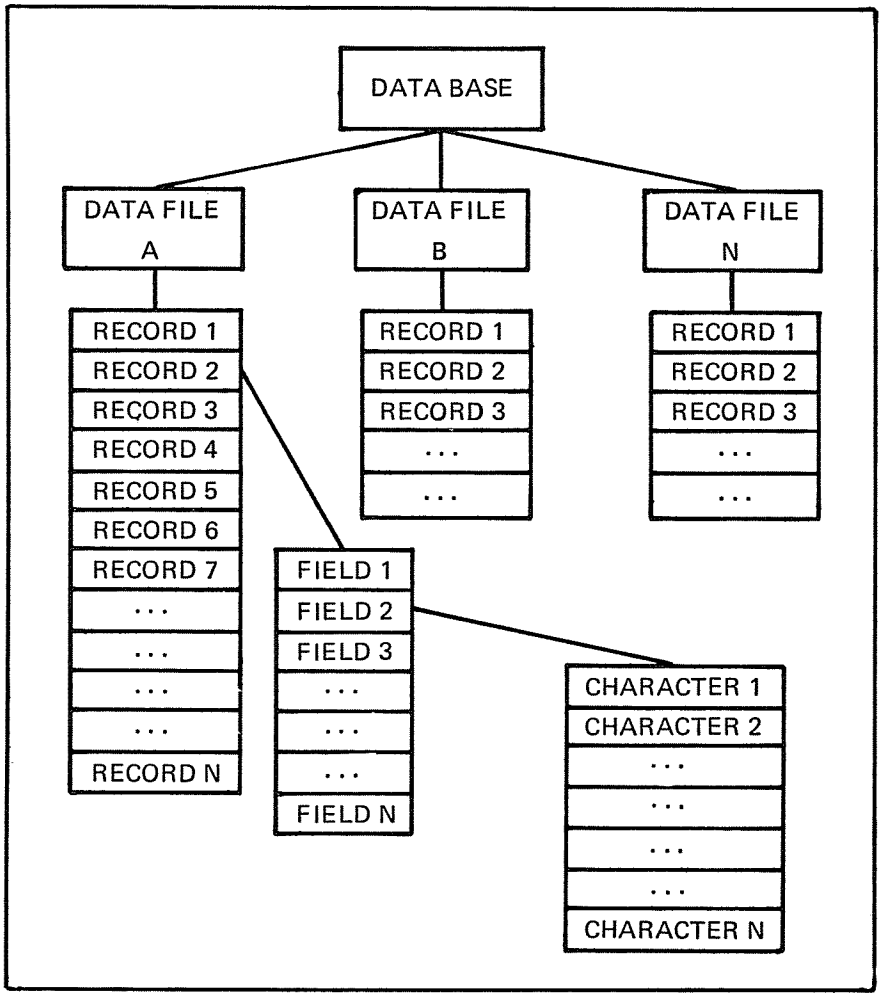
COMMANDS – are instructions given to the PIMS program to cause it to manipulate and analyze information for you. PIMS operates in two basic modes. One of these modes is the command mode. When in this mode the program accepts and interprets directives that pertain to how it is to act upon the information contained in the current data file. PIMS has commands that enable you to add data records to a file, to rubout information in a file and to change the information in a record. Other commands allow you to search for specific information in a field, to sort records by the contents of a field, and to sum the numerical value of a specific numeric field in a data file in several different ways. You can also tell PIMS to store data on an external storage device or to load data from such a unit. There is even a command that causes PIMS to help you by reminding you of all the command options that you have to work with! When PIMS is not in the command mode, it is in the data entry mode. In

the data entry mode it accepts data to be stored in the fields of records making up the current data file.

NAMES – are mnemonic words or symbols provided by the PIMS user to indicate the contents of fields. For instance, a field containing a person's name might simply be called the NAME field! A field containing an address might be named the ADDR field. If a record had two lines for person's address, the second address line might be named the ADR2 field. Names used for fields can be up to four characters in length.

## The Basics of Managing a Data Base

The key to the successful use of a data base management program such as PIMS lies in careful organization of the information that is to be processed and/or analyzed. A considerable portion of this publication is devoted to illustrating specific examples of methods that can be utilized. Before getting to examples, however, it will be



pointed out that to organize information properly, one needs a basic understanding of what kinds of operations a data base management program can perform. That is what we will talk about here.

*Data Is Organized as Records Within a File*

PIMS allows data to be organized as records within a file. Each record established in a data file is automatically assigned a reference number by PIMS. The first record entered in a data file becomes record number one. The second becomes record number two, and so forth. If a record is removed from a file, all the other records are automatically renumbered to account for the deleted record. If the records in a file are shifted in position within their file because of a sorting operation, they are all automatically renumbered to reflect their new positions within the file.

*A Record Is Composed of Fields*

A record may be composed of up to ten fields of information. A field is generally (but not always) thought of as all the information contained on a single display line. While two fields cannot be combined on one display line when using PIMS, it is possible for a field to be spread over more than one display line. This is because most displays automatically show data on more than one line when a certain number of characters have been placed on a line. PIMS will consider all the characters inputted to be in a field until it receives a "line enter" or "carriage return" character. Thus if a terminal automatically scrolls data to another line of the display for the sake of display clarity, the information will still be manipulated as part of a single field by PIMS – provided that the data has indeed been entered as being in a single field.

*PIMS Operates on the Contents of Fields*

Fields are important to PIMS users because PIMS essentially operates on the contents of fields in special ways. A record in PIMS is comprised of from one to a maximum of ten fields. All fields are assigned reference names by the user to identify their contents. Fields are also automatically assigned numbers by their order of establishment when a file is initially created by the operator. These numbers are sometimes used by PIMS to reference information in a record. Once fields have been established for a data file, PIMS can be directed to manipulate the contents of these fields in a variety of ways. This is done through the use of commands issued by the user.

*Searching a Data File*

PIMS can search a designated field in all the records of a data file for a specific entry – then show the user only the information contained in the records of interest. Thus, it can effectively screen out unwanted information.

*PIMS Can Form Sums*

PIMS can sum the contents of numeric fields in a data file. If, for instance, a group of records has a field that contains the

amount of a purchase, one can have the computer obtain a sum for that particular field over all the records, thereby providing the user with a total amount of all purchases recorded in the data file.

*It Can Even Do an "Intelligent Sum"*

What is more, PIMS can also do what we refer to as an "intelligent sum." You can tell PIMS to sum the contents of a specific numeric field for all the records in a data file that are identified by the specific contents of a second alphanumeric field. Thus, you could obtain a sum of all the purchases recorded in a file for a particular type of merchandise!

*Here's Real Power – Sort on a Field Capability!*

PIMS is able to sort the records in a file into an order determined by the contents of a designated field. You can sort a mailing list by zip code order. You could sort the same list using a different field by alphabetical order of names.

*Organize Your Sorts*

To use PIMS effectively you must pay close attention to the types of fields you establish for a particular application. If you want to be able to sort records by zip code number, you establish a field for that number alone. If you were to place the zip code information in a field that also contained the city and state of a person's address, you would not be able to sort in the manner desired. This is because PIMS always performs a sort by examining the data in a field from left to right. Putting the zip code information at the end of a field containing other data would result in the records being sorted as a function of the other data that preceded the zip code in the field.

*Editing Is Easy*

Records can be added, removed, or their contents altered. It is thus easy to keep information in a file up to date. If a person on your mailing list moves, you just change the address fields in the record that pertains to that individual.

*You Can Save Your Records*

Data files can be stored permanently on an external device, such as a magnetic tape system, so that when you turn your computer off your data base will still exist. When you want to use your data base again, you just turn your computer on and load the information back into your computer.

When you combine all the kinds of capabilities mentioned here into one convenient operating package (that is invoked at the touch of a few keys on a keyboard), you have PIMS, a powerful Personal Information Management System.

## Chapter 4

# Commanding PIMS

**Communicating with PIMS**

PIMS is an interactive program. This means that it operates in a conversational mode with the user. As it operates, it queries the user for commands that tell it what to do with the data it is accumulating or manipulating, or it asks that data be inputted. The user inputs data and commands via an input terminal. PIMS displays queries and information about the data it is processing on a video screen or hard copy printer at the terminal.

It is not at all difficult to learn how to communicate with and utilize PIMS. Really all you have to do is answer the questions that PIMS will ask you!

For instance, after PIMS has been loaded into your computer and placed into operation by executing a system RUN directive, it will initially display the following messages:

```
*COPYRIGHT (C) 1979 SCELBI PUBLICATIONS*
-----------------------------------------

ENTER OPERATION MODE:
1). CREATE NEW FILE  2). LOAD FROM CASSETTE
?
```

*How to Create a New Data File*

The question regarding the operation mode is really self-explanatory. If you already have a data file stored on a cassette that you wish to continue working with, you would respond with the number 2. If you were going to start building a new data file, such as would be the case when you first used PIMS, you would respond with the number 1.

As a rule of thumb, it is advisable to respond to any queries with just the information requested. Some BASIC interpreters are sensitive about the inputting of extraneous characters, even seemingly harmless ones such as blank spaces. Therefore, we recommend that you respond to queries directly. While PIMS does have a few subroutines that trap user errors, thus letting one get by with some minor input errors, the number of these subroutines is small and

they are not intended as a catch-all safeguard.

*Breaking Off*

If you should find that you have made a serious entry error after you have already depressed the carriage return or line entry key, you may abort an operation by returning to your system monitor using whatever method your system provides. For instance, on the Radio Shack TRS-80 level II, this may be accomplished by depressing the "BREAK" key. Once back in your system monitor you may either restart PIMS completely by issuing a RUN directive, which will wipe out any data base you may have established in the computer. Or, you may place PIMS back in operation without disturbing the current data base by entering the PIMS program at line 540. This is usually accomplished from a system monitor by inputting the directive: GOTO 540.

Notice: All entries to PIMS provided by the user, whether commands or data, must be terminated by depressing the carriage return or equivalent (line entry) key. This serves as a signal to the computer that you have finished typing the entry. The computer does not acknowledge a command or accept the data you type on a line until it detects the terminator character. The issuing of a carriage return or line entry character by the operator at the end of every entry will be assumed as an implicit part of every operation described in this manual. We will thus dispense with explicitly stating this requirement every time we describe an entry in this manual. If your system has line-editing capabilities, you can use this feature to your advantage. If you discover you have made a mistake on a line before you have terminated the line, you can backspace to correct your error(s)!

*"All entries . . . must be terminated by depressing the carriage return . . ."*

*How to Structure a Data File*

Once PIMS is in operation in a situation where the user is going to create a new data file, it is necessary to structure the new data file. This is done by giving mnemonic names to the various fields that will make up a record. A user may define up to 10 fields in a record. A field name may consist of one to four characters. (A space is considered a valid character in a field name.) Once defined, these names will apply to all the records in the data file as they are entered. Additionally, as the field names are specified, the user must indicate whether the field is to be restricted to holding numeric data, or whether it will store alphanumeric data. This field "type" designation is made by following the mnemonic name of the field by a comma and then the letter N (for a numeric field) or the letter A (for an alphanumeric field). Once a field name and type has been specified and entered into the computer it cannot be altered! If you do not like a structure you have created for a data file, you must start

the program over by going back to your system monitor.

The example provided here illustrates the type of dialogue that will take place when defining a data file structure using PIMS. (User entries are represented by bold type in this example.)

```
DEFINE FILE STRUCTURE; ENTER FIELD NAME,TYPE(A OR N)
 1. ? AMT,N
 2. ? DATE,A
 3. ? ADDR,A
 4. ? CODE,A
 5. ?
STRUCTURE DEFINITION COMPLETE
FIELDS ARE:
 1 : AMT   ,N
 2 : DATE  ,A
 3 : ADDR  ,A
 4 : CODE  ,A

(COMMAND):?
```

How do you tell PIMS that you are through defining field names? As indicated in line number 5 of the illustration, it gets the message when the user enters the line terminator character without having entered any information! (It will also cease asking for field names when 10 names – the maximum allowed – have been provided.) When either situation occurs, PIMS will display the field names and types for final review by the operator. It then goes into the command mode to await directives from the user. This is signified by display of the "command?" query.

What would happen if the user already had a data file stored on an external storage medium that was loaded into the computer in response to the initial dialogue with PIMS? The field assignments would automatically be set up by the computer to reflect the original structure of the data file when it was created.

*Use the HELP Command to Help Yourself*

Once PIMS is in the command mode, the user can direct the program to perform a variety of operations. Perhaps the first operation one should learn about is referred to as the HELP command. Typing in the letters HELP in response to a command query from the computer will result in PIMS displaying a list of available commands as a refresher for the operator's memory. Such a display is illustrated here as an introduction to all the commands available when using PIMS.

```
(COMMAND): ? HELP

VALID COMMANDS ARE:
ADD
LIST
CHANGE
LABELS
HELP
END
SEARCH
SUM
SAVE
RUBOUT
SORT
(COMMAND):?
```

These commands essentially do what they say they will do! However, we will spend a little time now to explain in detail what takes place when each type of command is invoked. You already know what the HELP command does. Here are details on the rest.

**The ADD Command Appends Records to a File**

ADD – This command is issued when you want to append records to the current data file residing in the computer. Immediately after it has been recognized, PIMS will display (in brackets) the number of the next field that will be appended to the current data file. It will then ask you to enter data for each of the fields you have defined for a record. It will do this by displaying the name you have assigned to each field and the field type (A or N) followed by a colon. It then expects you to type all the data that is to go into that field for the record being entered.

It is important when entering data to remember that if a character string contains a comma, you should precede the string with a quotation-marks character. Failure to do so will cause all of the data after the first comma in the line to be ignored by the program. Experience indicates that users are prone to forgetting about this requirement (which is dictated by the operation of the BASIC strings interpreter), especially when entering the "city, state" portions of an address, or when entering textual data.

You should also note that if you do not want to enter any data in a particular field, just depress the carriage return or line entry key to advance PIMS to the next field.

When all the fields in a record have been presented by PIMS, it will automatically advance to allow you to fill in the next record. As it does this, it will display the number of the new record in paren-

thesis before displaying the name of the first field in the new record.

*Stop When You Have Added Enough*

When you have entered data for all the records you desire to establish, and you want to return to the command mode of PIMS, you must enter the single word STOP into any field of a record. (Normally, this will be in the first field of a new record). Do not enter anything else in the field other than the word STOP! When the word STOP is recognized as the sole contents of a field when PIMS is in the append mode, it will cease accepting data and return to the command mode. It is important to note that if you use the STOP directive in any field of a record, all of the data in that record will be discarded!

**The LIST Command Displays Everything**

LIST – This command is used to display the contents of records. When the command is issued, PIMS will respond by asking whether it is to display all, just one, or a group of records. The letter A is entered by the operator if all records are to be displayed. PIMS then responds by listing the number of each record in the data file, followed by the field names, their types (alphanumeric or numeric), and their contents.

The letter O is entered by the operator if the contents of just a single record are desired. PIMS will ask "which one" meaning which record in the data file is to be displayed. The operator responds with the number of the desired record. PIMS will verify that number, then display the field names, their type, and their contents as requested.

The letter R is entered by the operator if the contents of a range of record numbers is to be displayed. If this is done, PIMS will ask for the lower bound, and then the upper bound, of the range of record numbers in the file that the operator wants to have displayed. PIMS then displays the contents of the records in that range, along with the associated information related to field names, etc.

**The CHANGE Command Allows Alterations**

CHANGE – This command is used when you wish to alter the contents of one or more records. When this command is issued, PIMS will respond by asking whether all, just one, or a whole group of records are to be changed. The user responds appropriately in the same manner as described for the LIST directive. Once a record has been specified as one that is to be altered, PIMS will access the record, display its number for verification, then display the contents of individual fields within the record. After each field has been displayed the operator may elect to alter the contents of a field, or

leave it as shown. If the contents of a field are to be altered, then the new contents of the field, *in its entirety,* is typed in on the keyboard. If the user does not wish to alter the contents of the field being displayed, then the hyphen (alternately referred to as the minus sign [−]) character should be entered by the operator. Note that if the operator does not enter any characters, and simply presses the carriage return or line entry key, that the contents of a field will be effectively cleared! We repeat. To leave a particular field unaltered, the hyphen (minus sign) character must be entered in response to the field change query!

If more than one record is to be altered, PIMS will automatically work through all the records requested, displaying each field in a record, and waiting for the operator to alter its contents, or approve of its original data.

It is important to realize that all of a record must be examined and approved on a field-by-field basis, even though possibly just one field is going to be altered! Make sure you get in the habit of using the hyphen to indicate that no changes are to be made to a field during the field review procedure!

**The LABELS Command Beats Typing**

LABELS – You use this command to print labels, such as gummed labels used for mailing purposes. The LABELS directive is like the LIST command with two main differences. First, because mailing labels are generally not very long, only the first five fields in a record are printed when data is outputted. Second, before the data is outputted PIMS provides an alignment procedure so that the printing machinery normally used for producing labels can be properly lined up to put the first data line at the top of a label.

During the alignment procedure, PIMS will print a single line and then wait for the operator to depress the carriage return or line entry key. This pause gives the operator the opportunity to adjust the label printing mechanism. When the key is depressed by the operator, PIMS will then print out four "dummy" lines consisting of the letter "X". These four dummy lines together with the previously printed line provide a printing mock-up that should fit within the area allowed on a label. This procedure is performed by PIMS three times to allow fine adjustments to be made to the printer mechanism. After the third mock-up label is printed the program will continue running to print out the first five fields of all the records requested. A blank line is issued between each record to allow for the space normally provided between labels on computer form-fed

sheets. At the conclusion of the label printing process PIMS returns to the command mode.

Sometimes it may be desirable to use the LABELS command to print labels other than those used for names and addresses. For instance, one might want to print labels to be used on spice bottles that contained just a single line of text. To align the text in the middle of a label, simply create a record with field numbers 1, 2, 4 and 5 given dummy names and left blank. Field number 3 can contain the name of the spice, or other information to be printed on the label. With a little thought, you can see that the LABELS command can be used in a variety of label-making applications.

**The SEARCH Command Helps You Find It**

SEARCH – This command is used to locate information in a data file. Of course, if a data file was small, you could pick out the information you needed just by listing out the contents of all the records in a file. However, when you are working with a large data base, the SEARCH directive becomes a much-appreciated convenience.

When you invoke the SEARCH command, PIMS will ask you the number of the field to be searched and the expression in that field that you are seeking. It will then search through *all* the records in the data file. Every time it locates a record in which the specified field contains exactly the expression sought *(and nothing else!)*, the entire contents of that record will be displayed along with the number of that record in the data file. (You may want to get in the habit of jotting down the record numbers as they come up. You will then know what records to call for using the LIST command if you want to study a record containing a lot of information. Remember that, if you are using a video display, records found by the search directive may be pushed off the screen as other records meeting the search criteria are located and presented to the operator.)

Note that you must specify the entire contents of the field that you wish to have searched. PIMS does not conduct sub-string searches within a field.

**The SUM Command Can Add All (Or Just Some) of it Up**

SUM – You will find this one of the most useful commands available in PIMS. The SUM command may be used to find the total numeric value of a specific field over all the records in a file. It may also be used to calculate what we call an "intelligent sum". That is a summing operation that takes place only over those records that meet your search criteria for a specified field. For instance, if you are maintaining a file that lists categories of items in one field, and the

price of an item in another field, you can have the computer total up the costs of all of the items of a particular category found in that file! Several examples of the SUM command will be provided later in this manual to demonstrate the power of the PIMS summation directive.

**The RUBOUT Command Takes it All Off**

RUBOUT – This command is used to eliminate unwanted records from a data file. When this command is issued, PIMS will ask whether all, just one, or a group of records are to be deleted. The user responds appropriately in the same manner as described for the LIST directive. Once a record has been removed by execution of this command, no trace of it will remain in the data file. All the remaining records in the file beyond the point where a record was deleted, will be renumbered automatically by PIMS to account for the removal of the record (or records). Incidently, the use of this command to eliminate all the records in a file is a convenient way to start a new data file, having the same structure as the one you were previously working on, when you are establishing a multi-file data base. (On the other hand, you should take extreme care when using this command to make sure you select the option desired when specifying whether all, just one, or a range of records are to be deleted!)

**The SORT Command Gets Things Straight**

SORT – Certainly the most powerful command provided on PIMS is the full alphanumeric and numeric sort capability. Using this command you can sort all of the records in your data file according to the alphabetical or numerical contents of any individual field. Just remember that the comparison algorithm used during the sort works on a left-to-right basis. Thus, if you wanted to sort a file according to a person's last name, you would have to order the name field so that the last name came first in the field. I.e., the name field would need to be ordered as Jones, Tom, rather than Tom Jones.

Note should be made of the fact that the sort routine operates on a character-by-character basis from left to right across a field. Ranking is made as soon as the N'th character in a field of a record is larger or smaller than the same N'th character in the comparison record. In systems such as the Radio Shack TRS-80 level II this means that numbers and most punctuation marks and symbols are ranked before (lower) than the letters of the alphabet. Thus, if a file has mixed data (letters and numbers), the records with numbers in the sort field will be arranged before those having strictly alphabetic data in the sort field.

When a file is sorted all of the records will automatically be renumbered by PIMS to reflect their new positions within the file. The new ordering must be utilized by the operator when specifying the range numbers of records to be listed, etc.

Several examples of the use of the SORT command are presented later in this publication. Be sure and review them so that you fully understand the operation of this powerful PIMS capability.

**The SAVE Command Puts Data Away**

SAVE – This command is used to direct PIMS to save all the records of the current file in memory on an external bulk storage device such as a magnetic tape cassette unit. Once saved on such a device, it can later be recalled into memory for further examination and manipulation – including the addition of more records to the file, or deletion of outdated records.

If you are using PIMS on a Radio Shack TRS-80 level II or similar system you can name the file written on the tape cassette. This is done by responding to the query provided by PIMS for a "label for the file being saved" with a user created name for the file. The file name you designate may then be used to locate that file on the tape and read it into memory at a later time. Most commonly used small computer systems allow for similar naming of bulk storage files. However, the exact procedure may vary slightly. Since the SAVE routine is one of several routines that essentially have to be customized for the particular type of system you are using, you should refer to the technical chapter in this book that describes the actual loading and implementation of PIMS. You should then also take into account the peculiarities of your own system when it operates with your external bulk storage device(s).

**The END Command Is a Terminator**

END – This command may be used to exit PIMS and return to your system monitor. When this directive is invoked, PIMS will first ask the operator whether the data in the current file has been saved. This query is made simply as a reminder to the user that the contents of the data file currently in memory may be altered or destroyed by the system's monitor program. If the user has forgotten to save data that was to be safeguarded, then PIMS will return to the command mode if a negative reply is made to the question. On the other hand, if the user does not care about saving the data in memory when the program is exited, the operator can tell PIMS a little white lie, by responding with a Y for "yes" to the query. Computer control then goes to the system's monitor program.

## Chapter 5

# 15 Typical Applications

**Building a Personal Mailing List**

The first application we have chosen to demonstrate is a task that can be implemented using just two PIMS commands: ADD and LABELS. We will show you how to create and use PIMS to establish and use a mailing list of friends, relatives and business contacts.

To start this data file we should decide just what type of information we want to store on the computer. Some obvious pieces of information that immediately come to mind are

1) Person's name

2) Person's address

However, in addition to this essential information, it might be nice to classify entries into categories based on the type of association one has with a person. Categories might include friend, relative, or business contact. We might also wish to store other kinds of information. How about providing room for a person's telephone number, a secondary address (such as a business associate's home address) and a line for general comments or notes?

You might have other kinds of information you would want to record for each entry. Fine. Just set aside a field in the record format you create for each kind of information that you want to be able to manipulate.

*Don't Store Too Much*

Be careful not to try and store too much information in a record. Remember, a record is limited to a total of about 240 characters (depending on how many fields are assigned to a record). To avoid having too much data stored in a data base, it is a good idea to practice identifying the most pertinent kinds of information you wish to store for a given application. It is also good to form the habit of estimating approximately how long a record will be as fields are created. This is accomplished by estimating the number of characters that a typical entry might contain.

For instance, in this application we might assume that the field for a person's name would typically contain twenty characters. In order to categorize a person's address, we might assign three fields

to different parts of the address. One field could hold the street address. The next could store the apartment number or post office box. The third could be reserved for the city and state portion of the address. Let's estimate that a typical entry would consist of approximately 25 characters in each of these three fields. A fifth field might be reserved for a person's zip code. Count on five characters in that field. Field number six in this application could hold the category type assigned for each person (friend, relative or business acquaintance). Count on, say, 10 characters as a typical entry in that field. Field number seven could store the individual's telephone number. Let's reserve 15 characters for that field in order to provide for area code and extension numbers. We might assume an abbreviated secondary address field entry would use about 50 characters as would a field for general comments. Adding up our estimates for this application we could say that we have used up about 225 characters of our allotted 240 per record. Time to stop creating fields!

Another factor to consider when creating a mailing list program is related to the operation of the LABELS command. Remember, when using that command, only the contents of the first five fields in a record will be printed on the labels produced by the program. We have taken that into account while structuring this example. Fields one through five contain addressing information only, i.e., that which will appear on the labels. The other fields will hold information that we do not plan to have printed out on mailing labels.

*Starting Your Dialogue With PIMS*

It is now time to show how we would interact with PIMS to establish and use our data in a mailing list application. For this first example, we shall assume that the user has just loaded the PIMS program into his or her computer and is ready to execute the program by issuing a system RUN directive. Also, to facilitate the reader's understanding at this point, the dialogue with the computer terminal will be shown with the user's entries in bold face type. This convention will be dropped after this initial example as the reader will readily observe that PIMS operates in a prompting mode. The user simply enters the appropriate information in response to the program's queries and prompts.

When PIMS is first placed into operation by executing your system's BASIC RUN directive, it will display an identifying header and copyright notice. It will then determine if you wish to establish a new data file from your keyboard or load in a previously created file from your external storage unit. In this example we are creating a new file so we respond appropriately to put PIMS into the key-

board entry mode.

*Creating a New Data File*

The first thing that must be done when creating a new data file is to establish and identify the fields that will be assigned to each record. This is accomplished by providing a mnemonic one to four character name for each field and then indicating whether the field will be alphanumeric or strictly numeric in content. The letter A designates an alphanumeric field, the letter N specifies a numeric field. Note in the following example of this procedure that the field mnemonic name must be followed by a comma before specifying the type of field (alphanumeric or numeric).

All keyboard entries to PIMS must be terminated by issuing a carriage return or line enter directive on the terminal. Since this procedure is universal with PIMS, it is not illustrated in any of the listings. Remember it!

```
RUN
*PIMS – PERSONAL INFORMATION MANAGEMENT SYSTEM*
*COPYRIGHT (C) 1979 BY SCELBI PUBLICATIONS*
------------------------------------------------------------

ENTER OPERATIONAL MODE:
1). CREATE NEW FILE  2). LOAD FROM CASSETTE
? 1
DEFINE DATA BASE STRUCTURE; ENTER FIELD NAME,TYPE(A OR N)
 1. ? NAME,A
 2. ? ADD1,A
 3. ? ADD2,A
 4. ? CS,A
 5. ? ZIP,N
 6. ? CAT,A
 7. ? TEL,A
 8. ? SECA,A
 9. ? COMT,A
10. ?
```

Note that we have given mnemonic names to the fields we have established for this example that correlate to the type of information each field will hold. We specified that the field named ZIP be numerical. All others are alphanumeric. We show ZIP being numerical for illustrative purposes only in this example. We could have set it to be alphanumeric for our mailing list application. The practical reason for establishing a field as numeric is that only numeric fields can be summed. There is no reason for us to want to sum zip codes in our mailing list application, but it hasn't done us any harm to specify the field as numeric since it will only contain numbers.

*"... only numeric fields can be summed."*

To terminate the process of defining the fields to be used in our records, we simply do not assign a name. Instead, we issue a carriage return or line enter directive on our terminal as shown in the line numbered 10 in the example. By the way, the numbers displayed by the computer correspond to the number of fields we have defined. They represent the internal references given to the fields by PIMS in addition to the mnemonic names we have specified.

When we have finished establishing our fields, the computer will respond by listing the field names we have created for our review as shown here:

```
STRUCTURE DEFINITION COMPLETE
FIELDS ARE:

1 : NAME ,A
2 : ADD1 ,A
3 : ADD2 ,A
4 : CS   ,A
5 : ZIP  ,N
6 : CAT  ,A
7 : TEL  ,A
8 : SECA ,A
9 : COMT ,A

[COMMAND] :? ADD
```

At the conclusion of this operation PIMS will be in the command mode as indicated by the display of the "command" query. It is important to note here that if for some reason we did not like the arrangement of our fields, we would have to stop the operation of the PIMS program by invoking the system's break directive (external to PIMS), then restart PIMS with a system RUN assertion.

*Adding Records to Your File*

Assuming we are satisfied with our fields arrangement, we can proceed to establish some records for our data file by giving the ADD command to our computer. In response to this request PIMS will display the number of the next record to be created in the data file. It will then present the name assigned to each field in a record, in the order in which they were initially assigned. In response to each name, the operator may enter the data desired in each field. This is how our creation of three sample records would appear on a terminal:

```
( 1 )
-----
NAME,A: MR. I. INTELLECT
ADD1,A: COLLEGE STREET
ADD2,A: APT 345
CS   ,A: SAN FRANCISCO, CA
ZIP  ,N: 87654
CAT  ,A: BUSINESS
TEL  ,A: (804) 343-1122
SECA,A: SILICON GULCH, CA 87653
COMT,A: MANAGER ENGR DIV., ACE PRODUCTS CORP.

( 2 )
-----
NAME,A: AUNT BESSIE
ADD1,A: BESSIE'S ORIGINAL COWFARM
ADD2,A: NEAR COUNTRY ESTATE
CS   ,A: OGDEN, UT
ZIP  ,N: 66554
CAT  ,A: RELATIVE
TEL  ,A: (123) 456-7788
SECA,A: NOT APPLICABLE
COMT,A: VISIT IN SPRING 79

( 3 )
-----
NAME,A: MS SLIM SHAPELY
ADD1,A: AT SORORITY
ADD2,A: BOX 234, STN. C
CS   ,A: CHARLOTTESVILLE, VA
ZIP  ,N: 22301
CAT  ,A: FRIEND
TEL  ,A: (704) 455-3232
SECA,A: RICHMOND, VA 22299
COMT,A: PERSONAL MATTERS

( 4 )
-----
NAME,A: STOP
END OF ADDITION
 3 RECORDS ADDED
[COMMAND]:? LABELS
```

When we wish to cease entering records to a file, we need simply type the directive STOP at the start of a field entry as shown in the example (in the NAME field of record number 4). PIMS responds to this directive by indicating how many records were added during the current operation. It then reverts to the command mode to await a new command.

*Printing Out Mailing Labels*

To have our mailing list printed out on labels, we need simply issue the LABELS command. (It is assumed here that a hard copy printer would be on-line to print the labels.) In response to this directive, PIMS queries as to whether we want to print all, one, or a range of records from our data file. In our example we respond with the request to print all the records in our file. PIMS then enters a "label alignment" procedure that enables us to make sure our label printing machinery is properly initialized. It issues an alignment message. The operator must respond to this message by depressing the carriage return or line enter key. PIMS then prints four more lines of filler material. The operator should check to see that all the printed material is centered on the mailing label in the printing machinery. This procedure is repeated a total of three times to allow the operator to make fine adjustments to the printer mechanism. PIMS then prints the contents of the first five fields of all the records that have been requested. At the conclusion of the label printing operation, it reverts to the command mode for further instruction from the system operator.

The sample listing shows the operator issuing the END command at the conclusion of the label printing operation. This directive causes the system to exit PIMS provided the user responds with a Y for yes to the question regarding the saving of the data file. This question serves simply as a reminder to operators to save data on an external storage device if they plan to use the data again in the future. Once PIMS is exited, the computer's operating system may destroy the data file placed in the computer's memory by PIMS.

```
ENTER MODE: A(ALL), O(ONE), R(RANGE)
? A
LOAD & ALIGN LABELS; HIT RETURN?
XXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXX

LOAD & ALIGN LABELS; HIT RETURN?
XXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXX
```

```
LOAD & ALIGN LABELS; HIT RETURN?
XXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXX

MR. I. INTELLECT
COLLEGE STREET
APT 345
SAN FRANCISCO, CA
87654

AUNT BESSIE
BESSIE'S ORIGINAL COWFARM
NEAR COUNTRY ESTATE
OGDEN, UT
66554

MS SLIM SHAPELY
AT SORORITY
BOX 234, STN. C
CHARLOTTESVILLE, VA
22301

[COMMAND] :? END

HAVE YOU SAVED ALL FILES PROPERLY?
ENTER Y/N? Y
END OF PERSONAL MANAGEMENT SYSTEM
A QUALITY SOFTWARE PRODUCT FROM SCELBI
```

**Balancing Your Checkbook**

We next consider a task that most people have to confront at least once a month. Balancing their checkbook. Learning to use PIMS and your computer to reconcile your checkbook with the bank's statement can save you valuable time. This is especially so if you are among the large group of people that experience end of the month surprises or difficulties when they attempt to perform the reconcilement. Let's face it. The computer can add and subtract much more accurately and faster than we can! Besides, why should we have to be bothered with such a mundane task?

*Organizing the Data*

Organizing our checkbook data to have PIMS do the dirty work for us is only a little more difficult than creating the personal mailing list file described in the preceding application. Let's take a look at the information we need to store to properly maintain a checkbook record. As far as expenditures go, we would want to record information such as

1) The date of a check

2) The check number

3) The name of the check recipient

4) The amount of the check

5) Purpose of the expenditure

On the other side of the ledger, keeping track of income to our checking account, we would very likely want to log such information as

1) The date of a deposit

2) The deposit number

3) Where we obtained the funds

4) The amount of the deposit

5) Notes or miscellaneous data

While you might be able to think of some other types of information that you would like to store in the computer regarding your checkbook, we will limit our current example to the kinds of information listed.

The reader might notice that we have already organized the data we desire to store into an efficient format. The information related to either expenditures (writing of checks) or receipts (deposits to the checking account) each fit nicely into five PIMS fields.

Since PIMS allows up to ten fields per record, one might consider letting each record contain all 10 fields. The appropriate group of fields would be filled out depending on whether one was storing an expenditure or a deposit. Such a scheme could indeed be implemented. However, a little study and thought shows that we could get by using just five fields. All we need to do is to establish the convention of using a minus value to record the amount of a check (item number 4 in the expenditures list) representing money being taken from the account. Item number 4 on the receipts list, deposits to the account, would be entered as positive values signifying an increase in the account! The remaining fields (1, 2, 3 and 5) may then serve for either deposits or expenditures as they can readily be interpreted in context with the data in field number four.

Here is how we could set up this application by assigning mnemonic names to five fields:

```
RUN
*PIMS – PERSONAL INFORMATION MANAGEMENT SYSTEM*
*COPYRIGHT (C) 1979 BY SCELBI PUBLICATIONS*
------------------------------------------------------------

ENTER OPERATIONAL MODE:
1). CREATE NEW FILE  2). LOAD FROM CASSETTE
? 1
DEFINE FILE STRUCTURE; ENTER FIELD NAME,TYPE(A OR N)
 1. ? DATE,A
 2. ? NUM,A
 3. ? DESC,A
 4. ? AMT,N
 5. ? COMT,A
 6. ?
STRUCTURE DEFINITION COMPLETE
FIELDS ARE:

1 : DATE ,A
2 : NUM  ,A
3 : DESC ,A
4 : AMT  ,A
5 : COMT,A
```

*Entering Transactions*

Once the fields have been identified we can enter our transactions on the computer. This could take place in a batch mode at the end of a month. Or, entries could be made on a daily basis, depending on the activity of your account. Some sample entries are shown here:

```
[COMMAND] :? ADD

( 1 )
-----
DATE ,A: DEC0178
NUM  ,A: D101
DESC ,A: SALARY
AMT  ,N: 1400
COMT ,A: SAVINGS

( 2 )
-----
DATE ,A: DEC0278
NUM  ,A: C214
DESC ,A: DEPOSITED TO SANDRA'S ACCT
AMT  ,N: –850.00
COMT ,A: MONTHLY ALLOWANCE
```

```
( 3 )
-----
DATE ,A: DEC0378
NUM  ,A: C215
DESC ,A: VERMONT SKI LODGE
AMT  ,N: -115.50
COMT ,A: DEPOSIT ON SKI TRIP

( 4 )
-----
DATE ,A: DEC0878
NUM  ,A: D102
DESC ,A: CONSULTATION FEE FROM ERDA
AMT  ,N: 2200.00
COMT ,A: TO BE DECLARED IN 78

( 5 )
-----
DATE ,A: DEC1278
NUM  ,A: C216
DESC ,A: HOKIE FLYING CLUB
AMT  ,N: -129.95
COMT ,A: NOV DUES & CHARGES

( 6 )
-----
DATE ,A: DEC1678
NUM  ,A: C217
DESC ,A: PLAYBOY MAGAZINE
AMT  ,N: -28.00
COMT ,A: GIFT SUBSCRIPTION

( 7 )
-----
DATE ,A: STOP
END OF ADDITION
 6 RECORDS ADDED
[COMMAND] :? CHANGE
```

*Here's How to Change a Record*

We will take the opportunity here to introduce the reader to the practical application of another PIMS command – the CHANGE directive.

Suppose we discovered after we had made entry number 6 in the example that the amount of the check we had recorded was incorrect. We would invoke the CHANGE command to access record number 6 and make a corrected entry to the selected field in the manner shown. Note in the example that if we do not wish to alter the contents of a field, that we enter a hyphen mark (–) immed-

iately following the question mark on the line following the field's current contents. If we did not enter the hyphen mark, and simply pushed the carriage return or line enter key, then the contents of that field would be effectively erased! We repeat: If you do not want to change the contents of a specific field when using the CHANGE directive, you must enter a hyphen sign (often used as the minus sign on many terminals) before depressing the carriage return or line enter key.

```
ENTER MODE: A(ALL), O(ONE), R(RANGE)
?O
WHICH ONE? 6
( 6 )
-----
DATE ,A: DEC1678
?-
NUM  ,A: C217
?-
DESC ,A: PLAYBOY MAGAZINE
?-
AMT  ,N: -28.00
?-25.00
COMT ,A: GIFT SUBSCRIPTION
?-
 6
[COMMAND] :? SUM
```

*Reconciliation Using the Summing Capability of PIMS*

Now, how do we use PIMS to perform reconciliation, the really essential task in this application? We utilize the SUM command to obtain a total of all the entries in field number 4 – the AMT (amount) field! PIMS will perform the summation process, deducting the negative entries representing checks from the positive values representing deposits and reporting the results in the manner illustrated.

```
ENTER FIELD NUMBER TO BE SUMMED? 2
IMPROPER FIELD TYPE; ENTER AGAIN
ENTER FIELD NUMBER TO BE SUMMED? 4
DO YOU WANT TO SUM OVER ALL RECORDS?(Y/N)? Y
( 1 )( 2 )( 3 )( 4 )( 5 )( 6 )( 7 ) SUM OF AMT  ,N = 2479.55
[COMMAND] :? END

HAVE YOU SAVED ALL FILES PROPERLY?
ENTER Y/N? Y
END OF PERSONAL INFORMATION MANAGEMENT SYSTEM
A QUALITY SOFTWARE PRODUCT FROM SCELBI
```

Presto! Your work is done. By the way, in a practical situation you would want to initialize each file with a record indicating your current account balance (amount on hand or balance forwarded)!

**Organizing Recipes**

In the preceding applications we have had the opportunity to demonstrate the use of the ADD, LABELS, CHANGE and SUM commands. We shall next consider an application that will enable us to show the capability of the SEARCH command. We will also introduce the LIST directive.

Our goal in this application is to store cooking recipes in such a manner that we will be able to find a desired recipe quickly and efficiently. We no longer want to be bothered with looking through kitchen drawers for scraps of paper, or going through unwieldy file cards, or bothering with recipe books. We want something nice and simple, such as the capability of punching a few buttons on our computer and having the desired recipe displayed in a matter of seconds!

Since an object of presenting these examples of applications for PIMS is to help the reader learn how to organize information when using a data base management program, we are going to suggest that you stop reading for a few moments. Think about how you might organize data for a recipe file using the capabilities of PIMS. Why not write your method down on a piece of paper? When you have outlined your method of organization, continue reading to compare it with the example we will illustrate. Who knows, your method may be better!

*Condensing Data Using Coding*

For the purposes of our illustration, we are going to assume that the data to be stored will consist of the recipe, the most essential ingredient used in the recipe, and the number of people the basic recipe is designed to serve. We plan to isolate the latter two pieces of information in a field separate from the recipe field so that we can locate categories of recipes quickly and efficiently. We plan to use the search capability of PIMS to find those types of recipes that we might be interested in at any given time. To facilitate the searching process, we will use a technique for condensing information which everyone is familiar with whether they have used a computer or not. It is called coding. PIMS users will soon learn that certain types of information can be located (or sorted, as will be explained later) more readily if the information is condensed by coding. A great deal more data can be processed in a small computer system through the use of this technique.

How can coding help us in our present application? Suppose we categorize our recipes' most essential ingredient by the type of meat utilized? We further categorize a recipe by the number of people it is designed to serve. Now, instead of entering the full information for the first type of classification such as beef, chicken, fish, lamb, etc., we could condense this information by simply using the first letter of the type of meat. Thus, one would have the codes B, C, F, L and so forth. (This coding scheme assumes that no two types of meat start with the same letter.) Next, the coding for the number of people a recipe serves could be added in to yield combined codes such as B2, C4, F1 and L6. Expanded, these codes would mean beef recipe for two people, chicken for four, fish for one, and lamb for six, respectively.

One can save a lot of memory space in a small computer system using this type of coding technique. "Chicken recipe for four" has twenty characters in it (not including spaces). Those twenty characters reduce to just two when the information is coded as "C4". That is a ten to one reduction in the amount of memory space used!

Coding also has advantages when it comes to searching (or sorting) data. Searching for the message "chicken recipe for four" will take considerably longer than looking for "C4" because the computer must search on a character-by-character basis. Not only does it take longer, but the chances for operator error are increased when the comparison string is lengthened. Remember, a computer must look for exactly the right sequence of characters when performing a search. If an operator mistakenly tells the computer to search for the phrase "chicken recipe for for" the computer will search all the records without finding the desired entry – because the last word of the search string should have been "four"! Coding entries can greatly reduce the chances of operator error in such cases.

You should plan on looking for ways to code data as you utilize PIMS. You will find it a considerable boon to your effective utilization of your computer's capabilities.

*Using Two Records for a Single Recipe*

Sometimes we might find that a recipe will need more room than is provided in a single PIMS record. Is there a way we can spread the recipe over several records? Sure – again through a form of coding! We can assign a field to each recipe that we will consider as our own numbering or paging system. We can then assign a recipe number to each recipe we enter (independent of the fact that PIMS assigns a number to each record in a data file). We can further

encode our own personal numbering system so that it indicates the page number of a multi-page entry (considering each record used to be a page). Thus, if our recipe number two is spread over two pages, we could assign our recipe numbers as 2.1 and 2.2 – meaning recipe number two, sections one and two respectively! Later you will see how this type of coding might be particularly valuable if you sort your records into different kinds of orders. With a field dedicated to this coded information, you can always get your entries back into their original order by sorting on that field.

The accompanying listing illustrates how our entries of three different recipes might appear as they are stored in four records. (Recipe number two is a multiple-record entry.)

```
( 1 )
-----
CODE ,A: C2
NUM  ,A: 1.1
RECP ,A: THIS RECIPE OBTAINED FROM SISTER. TAKE NO MORE THAN ONE
HALF HOUR TO GET READY. THEN WALK OVER TO COLONEL SAUNDERS. BUY
JUMBO DINNER. COME HOME. SERVE ON PLATE.

( 2 )
-----
CODE ,A: B4
NUM  ,A: 2.1
RECP ,A: CUBE MEAT. SAUTE IN BUTTER. ADD 1/2 TSP SALT AND 2 TBSP
OF WORCESTERSHIRE SAUCE. SIMMER HALF HOUR TURNING CUBES OCCA
SIONALLY. ADD CHOPPED ONIONS TO TASTE. BLEND IN TOMATOE PASTE.
SIMMER 3/4 HOUR MORE. SERVE OVER HOT RICE.

( 3 )
-----
CODE ,A: B4
NUM  ,A: 2.2
RECP ,A: FOR VARIATION ADD PARSLEY FLAKES AND BAY LEAVES. FOR
HOTTER SAUCE ADD MEXICAN PEPPERS AND TOBASCO SAUCE PLUS REGU
LAR PEPPER. WARN GUESTS!

( 4 )
-----
CODE ,A: H2
NUM  ,A: 3.1
RECP ,A: SLICE PRECOOKED HAM INTO 1/4 INCH THICK SLICES. BROWN ONE
SIDE IN FRYING PAN. TURN OVER. SPRINKLE TOP WITH BROWN SUGAR AND
CLOVE POWDER. BROWN REMAINING SIDE. SERVE WITH PINEAPPLE SLICE.
```

*Using the LIST Command*

If we wanted to get a copy of all the recipes stored in our PIMS recipe data file at any given time, we could utilize the LIST command. This command will cause all the records in a data file to be displayed on the terminal or hard copy printer.

*Finding It with the SEARCH Command*

However, in a typical situation, you would only be interested in a particular recipe. You could use the SEARCH command to get the desired recipe in several ways. If you already knew, for instance, that the recipe you wanted was stored as your number 3.1 in the data file, you could invoke the SEARCH command in the manner shown:

```
(COMMAND):? SEARCH

ENTER FIELD NO. TO BE COMPARED? 2
ENTER EXPRESSION TO BE COMPARED? 3.1
( 4 )
-----
CODE ,A: H2
NUM  ,A: 3.1
RECP ,A: SLICE PRECOOKED HAM INTO 1/4 INCH THICK SLICES. BROWN ONE
SIDE IN FRYING PAN. TURN OVER. SPRINKLE TOP WITH BROWN SUGAR AND
CLOVE POWDER. BROWN REMAINING SIDE. SERVE WITH PINEAPPLE SLICE.
```

If, on the other hand, you did not know the number of the recipe in your file, but you did know that you wanted a beef recipe that served four people, you could invoke the SEARCH command in the manner shown here:

```
(COMMAND):? SEARCH

ENTER FIELD NO. TO BE COMPARED? 1
ENTER EXPRESSION TO BE COMPARED? B4
( 2 )
-----
CODE ,A: B4
NUM  ,A: 2.1
RECP ,A: CUBE MEAT. SAUTE IN BUTTER. ADD 1/2 TSP SALT AND 2 TBSP
OF WORCESTERSHIRE SAUCE. SIMMER HALF HOUR TURNING CUBES OCCA
SIONALLY. ADD CHOPPED ONIONS TO TASTE. BLEND TOMATOE PASTE.
SIMMER 3/4 HOUR MORE. SERVE OVER HOT RICE.
```

```
( 3 )
-----
CODE ,A: B4
NUM  ,A: 2.2
RECP ,A: FOR VARIATION ADD PARSLEY FLAKES AND BAY LEAVES. FOR
HOTTER SAUCE ADD MEXICAN PEPPERS AND TOBASCO SAUCE PLUS REGU
LAR PEPPER. WARN GUESTS!
```

**Maintaining Maintenance Records**

Our next application is one which may currently be more applicable to business than to personal life. However, with more and more appliances and machines finding their way inside the home, this situation may soon change. That is one reason why we are presenting this example application here.

Suppose you wanted to keep track of all maintenance procedures and the related expense on anything you owned for a period of one year. This record keeping would include the disposition of everything of value. Thus, it might make note of burned out light bulbs that were replaced. The repairs made on your car. The tax deductible value of items such as old clothes given to charity. And, the various odds and end of small part bought during the year, such as a new doorknob for the back screen door, etc. You might also use this data file to build up a history on the performance of various appliances.

How much time would it take to keep an orderly record of this kind of information? Not very much – if you utilized PIMS once or twice a week (or month, depending on how often you have to repair things around your house). You would simply add records to your data file which could then be processed to yield information. Information that might surprise you! Such as how much all those repairs actually cost you.

Here is how you might assign fields to hold the kinds of information to be stored in such an application.

```
DATE OF REPAIR (YR-MO-DY):
ITEM DESCRIPTION:
REPAIR ACCOMPLISHED:
COST OF REPAIR:
CODE FOR SORTING:
COMMENTS:
```

For coding purposes you might use the following one-letter classifications:

```
A = AUTOMOBILE EXPENSE
C = CHILDREN'S EXPENSE
G = GENERAL HOUSEHOLD ITEM
T = THROWN AWAY/REPLACED
R = REPAIRED
```

A typical entry in the data file might appear as

```
DATE ,A: 78-10-01
ITEM  ,A: MOUSETRAP
REP   ,A: CHEESEHOLDER FIXED
COST  ,N: 1.43
CODE  ,A: G
COMT  ,A: WARRANTY EXPIRED
```

Suppose you had seven items in your data file at the end of the first week of keeping such records. You are beginning to get suspicious that this repairing business is taking more of your hard earned bread than you had realized. You could quickly issue the SUM command to get a total of expenditures from field number 4 representing the cost of repairs. Suppose this activity went as follows:

```
(COMMAND):? SUM

ENTER FIELD NUMBER TO BE SUMMED? 4
DO YOU WANT TO SUM OVER ALL RECORDS?(Y/N)? Y
( 1 )( 2 )( 3 )( 4 )( 5 )( 6 )( 7 ) SUM OF COST,N =  133.58
```

Are you spending too much money on household repairs? Is it worth having a computer and a program like PIMS to help you discover those hidden repair costs? You now know that PIMS can help you manage the situation better in the future!

**Maintaining an Inventory of Household Valuables**

How can PIMS maintain an inventory of household items? The details of interest for entry in an inventory file are

```
NAME OF AN ITEM
COST OF AN ITEM
DATE OF PURCHASE
WHERE PURCHASED
CODE FOR SORTING
```

The first four fields are similar to ones we have used in earlier applications. In this and some other applications, we have assigned a CODE field. We have done so with an eye toward facilitating the proper use of the SEARCH and SORT commands. For small data files, say, 20 to 50 records, it is perhaps unnecessary to group the inventory items. If you ever needed to recall an item, you could scan the whole file without much effort. Our objective, however, is to learn to organize information efficiently so as to take care of not only our immediate needs, but future needs as well. It would certainly be time consuming, even on the computer, if you had to look through 50 or 100 items to find only a few that were of immediate interest. Grouping of items might be by cost of the item or by room number in which that item was stored. You may even develop a code similar to the kind we used in classifying recipes that will describe the cost and its in-house location along with, perhaps, some other information. We shall leave that to you. In our example we shall simply use a one letter code to describe the in-house location of the item. Here are the codes:

```
B = BEDROOM
L = LIVING ROOM
A = ATTIC
G = GUEST ROOM
D = DOWNSTAIRS (BASEMENT)
```

We shall make all fields alphanumeric except the COST field. If you have been putting off making such an inventory list until now, you may prepare it easily while you duplicate this application of PIMS as an exercise on your computer. A sample entry follows:

```
NAME ,A: BEDROOM SUITE
COST ,N: 2000.00
DATE ,A: JUNE77
SHOP ,A: BASSETT GALLERIES
CODE ,A: B
```

Want to know what your inventory of possessions is worth? Here is the procedure:

```
(COMMAND):? SUM

ENTER FIELD NUMBER TO BE SUMMED? 2
DO YOU WANT TO SUM OVER ALL RECORDS?(Y/N)? Y
( 1 )( 2 )( 3 )( 4 )( 5 )( 6 )( 7 )( 8 )SUM OF COST,N= 12070
```

The above printout indicates that you had a total of 8 inventory records and the sum of the cost is $12,070.

## Putting Your Personal Mailing List to Work

An earlier section presented the straight-forward creation of a mailing list. On a small list, it is possible to perform a variety of tasks like sorting or grouping just by visual inspection. You could count the number of friends vs. the number of relatives simply by counting them as they appear on your video terminal. That would not be feasible as the number of entries in your mailing file approaches 100. At times you may wish to organize the entries by zip code order so you could derive a quick frequency distribution by state. Or, you might want to make labels for just a certain category of entries on the list. PIMS is ideally suited for performing such chores and we shall spend this and the next several sections in illustrating how.

The command that can organize your entries by zip code or by any alphabetic category is SORT. To demonstrate this capability, let us use a file with records structured like this:

```
NAME OF PERSON
ADDRESS
CITY, STATE AND ZIP
ZIP
CATEGORY (FRIEND, RELATIVE, BUSINESS CONTACT)
```

The data has been entered in random order. Initiating the LIST command provides the following printout:

```
NAME ,A: MR. MAGOO
ADR1 ,A: 2 FUNNY STREET
ADR2 ,A: SOMECITY, CA 91234
ZIP  ,N: 91234
CAT  ,A: RELATIVE

NAME A: MR. PRESIDENT
ADR1 ,A: PENN SYLVANIA AVE
ADR2 ,A: CAPITOL CITY, USA 11111
ZIP  ,N: 11111
CAT  ,A: NEIGHBOR

NAME ,A: MARY TYLER
ADR1 ,A: STAR ROUTE
ADR2 ,A: HOLLYWOOD, CA 12345
ZIP  ,N: 12345
CAT  ,A: NEIGHBOR
```

Now, suppose you wanted to put all of the enteries in your list into zip code order. Through use of the SORT command, the records can be organized into ascending order. Here is the procedure:

```
(COMMAND):? SORT
ENTER FIELD NUMBER TO BE SORTED? 4
FIELD FOR SORT IS: ZIP  ,N
DO YOU WANT TO CHANGE IT? (Y/N)? N
*** EXTRACTING KEY FIELD ***
*** START OF SORT ***
*** END OF SORT ***
(COMMAND):? LIST
```

Note that you were asked the question, "Do you want to change it?" It is asked to avoid sorting the wrong field if an incorrect number has been entered. The list, now sorted in zip code order, appears below:

```
NAME A: MR. PRESIDENT
ADR1 ,A: PENN SYLVANIA AVE
ADR2 ,A: CAPITOL CITY, USA 11111
ZIP  ,N: 11111
CAT  ,A: NEIGHBOR

NAME ,A: MARY TYLER
ADR1 ,A: STAR ROUTE
ADR2 ,A: HOLLYWOOD, CA 12345
ZIP  ,N: 12345
CAT  ,A: NEIGHBOR

NAME ,A: MR. MAGOO
ADR1 ,A: 2 FUNNY STREET
ADR2 ,A: SOMECITY, CA 91234
ZIP  ,N: 91234
CAT  ,A: RELATIVE
```

PIMS can sort your mailing list by other fields, too. For example, field 5 (category) could be sorted so that all names in each category will be grouped together. When using the SORT command on an alphanumeric field, the records will be placed in alphabetical order.

**Keeping Track of Department Store Charges**

In this application, we shall introduce the use of the intelligent SUM. This feature of PIMS allows the user to sum over only a selected number of records, instead of all the records as in the ordinary SUM.

This is accomplished by performing the summation for only those records, for example, which have identical information in another field. The optimum use of the intelligent SUM requires some additional care in the structure of files and data entry. This will be demonstrated best by a working example.

Suppose you have charge accounts in various stores. You find the grand total of their end-of-the-month bills to be more than the amount you had allocated in your budget. To avoid future overspending you would like to know the amount you have charged at each of the stores prior to making any new purchase at that store. Yes, you received the sales receipts for each purchase but, "they are in various places now," you say, and, "it would take an undue amount of time to find and add them up." PIMS can help. All you have to do is enter each charge before you lose that receipt or forget that purchase. The fields needed to open the file are as follows:

```
STORE NAME:
DATE OF PURCHASE:
AMOUNT OF PURCHASE:
DESCRIPTION OF PURCHASE:
```

You may add other fields if necessary. But, the above number is sufficient for our purpose. The AMNT (amount) field will be numeric. All others are alphanumeric. The following shows the entry of data into these fields and an ordinary summing of the amounts of purchase as shown in each field 3.

```
(COMMAND):? ADD

( 1 )
-----
STOR ,A: MACY'S FIFTH AVENUE
DATE ,A: 111278
AMNT ,N: 38.00
DESC ,A: HANDCRAFTED PURSE

( 2 )
-----
STOR ,A: DOC'S CANDY STORE
DATE ,A: 111478
AMNT ,N: .75
DESC ,A: REESE CUPS
```

```
( 3 )
-----
STOR  ,A: LEGGETT'S UPTOWN DRUGS
DATE  ,A: 111678
AMNT  ,N: 67.95
DESC  ,A: WESTERN JACKET

( 4 )
-----
STOR  ,A: MACY'S FIFTH AVENUE
DATE  ,A: 111878
AMNT  ,N: 2.50
DESC  ,A: NAIL POLISH

( 5 )
-----
STOR  ,A: MACY'S FIFTH AVENUE
DATE  ,A: 111978
AMNT  ,N: 29.34
DESC  ,A: COCKTAIL SHOES

( 6 )
-----
STOR  ,A: DOC'S CANDY STORE
DATE  ,A: 112478
AMNT  ,N: .78
DESC  ,A: BUBBLEGUM

( 7 )
-----
STOR  ,A: STOP
END OF ADDITION
 6 RECORDS ADDED
(COMMAND):? SUM

ENTER FIELD NUMBER TO BE SUMMED? 3
DO YOU WANT TO SUM OVER ALL RECORDS?(Y/N)? Y
( 1 )( 2 )( 3 )( 4 )( 5 )( 6 ) SUM OF AMNT  ,N = 139.32
```

Now, to demonstrate the use of the intelligent SUM, let us say that we wish to know the total amount for all purchases at Macy's Fifth Avenue:

```
(COMMAND):? SUM
ENTER FIELD NUMBER TO BE SUMMED? 3
DO YOU WANT TO SUM OVER ALL RECORDS?(Y/N)? N
ENTER FIELD NO. TO BE COMPARED? 1
ENTER EXPRESSION TO BE COMPARED? MACY'S FIFTH AVENUE
( 1 )( 4 )( 5 ) SUM OF AMNT  ,N = 69.84
```

Purchases at Macy's Fifth Avenue were found in records 1, 4 and 5 for a total of $69.84. It is important that all the purchases at this store were entered with consistency. Had one of the entries been made, for example, with the name Macy's 5th Avenue instead of Macy's Fifth Avenue, the former record would not have been counted in the intelligent SUM.

**Accounts Receivable**

Suppose you are the owner of a small retail store selling cameras and photographic supplies. Most of the sales are from walk-in customers and paid for in cash or by credit card. But some special customers are allowed to put purchases on account. Conventional systems for keeping track of such accounts receivable with paper work can be quite laborious. PIMS can be used to provide considerable convenience for small-scale receivables like this. It can quickly generate a report on the total outstanding amount at 60 days, 90 days or whatever. You can also conveniently prepare a summary of an individual customer's account. Here is how to set up the data base:

```
DATE OF SALE (YR-MO-DY):
TOTAL AMOUNT:
ITEMS:
LINE 4 NOT USED:
LINE 5 NOT USED:
CUSTOMER'S NAME:
ADDRESS:
CITY, STATE AND ZIP:
MONTH OF PURCHASE:
```

A typical entry would look like this:

```
DATE ,A: 79-01-01
AMT  ,N: 325.00
ITMS ,A: SUPERBROWNIE 500
SKIP ,A:
SKIP ,A:
NAME ,A: CHARLIE SHUTTERBUG
ADR1 ,A: 35 MILLIMETER DRIVE
ADR2 ,A: "MONOCHROMATIC, AK 99999"
MOP  ,N: 01
```

When a sale on account is made, the foregoing data is entered. When payment is received, use the RUBOUT command to delete the record. Only currently outstanding items are thus kept in the computer file. Once all of your data is entered into the computer, you can begin to extract the type of info needed for planning and ad-

ministration.

You may wish to know, for example, the total amount receivable from all customers which is 60 days past due. Assume for this example that the present month is June. Thus, April (or 04) is the month in which the 60 days past due sales were made. It is necessary, therefore, to sum field 2 (amount) for those records which have 04 (April) in field 9 (month). The answer can be obtained quickly with this procedure:

```
(COMMAND):? SUM
ENTER FIELD NUMBER TO BE SUMMED? 2
DO YOU WANT TO SUM OVER ALL RECORDS? (Y/N)? N
ENTER FIELD NUMBER TO BE COMPARED? 9
ENTER EXPRESSION TO BE COMPARED? 04
( 1 )( 2 )( 3 )( 4 ) SUM OF AMNT,N = 2300.00
```

The overdue amount for 60 days is $2300.00. The amounts for 90 or 120 days could be obtained by repeating the procedure and substituting 03 (March) or 02 (February) in field 9.

A statement of an individual customer's account is another valuable function of an accounts receivable system. It is used for preparing invoices, making decisions regarding credit limits or initiating steps for collection. Here is the procedure for getting a statement of a customer's account:

1) Sort records by field 1 (date).

2) Sort by field 6 (customer's name).

3) LIST and observe the range of record numbers occupied by this customer.

4) Using the LABELS command, ask for a print out of this range of records. It will produce a summary of purchases made by the customer which will be in the following format:

```
79-01-01
325.00
SUPERBROWNIE 500

79-01-02
400.00
1 CASE TOADACOLOR FILM

79-01-03
9.95
SCELBI'S ANTENNA HANDBOOK
```

## Sales Analysis

What salesman is doing the highest dollar volume? For the purpose of commission computation, what was the salesman's June sales total? What were the total sales last month for product A? These are key questions in sales analysis. PIMS has the capability of providing the answers. In this example, you are a distributor with five salesmen each selling three products. Every week the salesmen give you a summary of their performance by product. Here is the entry format:

```
MONTH:
WEEK OF THE MONTH:
SALESMAN'S NAME:
PRODUCT A SALES:
PRODUCT B SALES:
PRODUCT C SALES:
TOTAL SALES:
NUMBER OF CALLS:
```

A typical entry on a computer would look like this:

```
MO    ,N: 07
WEEK ,N: 01
NAME ,A: HARRY HUSTLE
PROA ,N: 7000.00
PROB ,N: 1000.00
PROC ,N: 9.95
TOT   ,N: 8009.95
CALS ,N: 25
```

At the end of the month when computing commissions, the total sales volume for Harry Hustle can be computed as follows:

```
(COMMAND):? SUM
ENTER FIELD NUMBER TO BE SUMMED? 7
DO YOU WANT TO SUM OVER ALL RECORDS? (Y/N)? N
ENTER FIELD NUMBER TO BE COMPARED? 3
ENTER EXPRESSION TO BE COMPARED? HARRY HUSTLE
( 1 )( 2 )( 3 )( 4 ) SUM OF AMNT,N = 40009.95
```

The procedure for computing the month's sales for product A is similar to the foregoing. In this case, field 4 is summed over all records. To determine which salesman did the highest volume with product A, use the SORT command and enter field 4 as the field number to be sorted. Then LIST the records and observe the ranking. Who did the highest volume with product A during the second week of July? Follow this procedure to find the answer:

1) Sort records by field 4 (product A sales).

2) SEARCH entering 2 (week of the month) as the field number to be compared and 02 (for the second week) as the expression to be compared.

3) LIST and observe the ranking. It will indicate which salesman did the highest volume with product A.

**Building a Private Reference Library**

How many times when in a discussion, conversation or an arguement, did you recall reading something in a magazine that supported your point of view. You try to find the magazine from the stack that contained the article only to be frustrated by the incompleteness of your memory. You can't locate the article. A lot of people read maybe six or seven magazines a month. Some tear off those pages they are likely to make use of later. This practice, of course, greatly reduces for others the usefulness of that issue. In a family or work community environment, it is a habit to be discouraged. The ideal solution would be to make a list of such articles, along with the pertinent contents, so that the data may be retrieved later, when desired. An arrangement that can be easily maintained with the help of PIMS is listed here:

```
CATEGORY OF ARTICLE:
TITLE OF THE ARTICLE:
PUBLICATION IN WHICH IT APPEARED:
INFORMATION OF INTEREST:
```

Each member of the family may have his or her personal reference file, or there may be one master reference file. In the latter case, an appropriate code must be added to describe the member of the family to whom that information is of interest. With conciseness you should be able to condense all the pertinent information into just one record. But, if necessary, more than one record may be used for a reference as in the recipe file example. Here is a sample of a private reference library data base:

```
CATG ,A: CHILD CARE
TITL ,A: ALTERNATIVE TO DR SPOCK
PUBL ,A: TIME DEC 23 78
INFO ,A: OTHER WAYS TO KEEP BABY FROM BEDWETTING

CATG ,A: MARRIAGE
TITL ,A: ROMANCE AFTER 70
PUBL ,A: COSMO OCT 78
INFO ,A: KEEP FOR FUTURE REFERENCE
```

```
CATG ,A: CHILDREN
TITL  ,A: CHILDREN'S RIGHTS AMENDMENT
PUBL  ,A: TINY TOTS TABLOID AUG 79
INFO  ,A: AN IDEA WHOSE TIME HAS COME

CATG ,A: CHILD CARE
TITL  ,A: SUMMER DISHES FOR THOSE IN DIAPERS
PUBL  ,A: GOURMET GUIDE MARCH 79
INFO  ,A: SHOW THEM THAT YOU LOVE THEM
```

If you are looking for an article on the subject of child care, use the SEARCH command to examine the category field (field number 1) and display all enteries for your review.

```
(COMMAND)? SEARCH

ENTER FIELD NUMBER TO BE COMPARED? 1
ENTER EXPRESSION TO BE COMPARED? CHILD CARE
( 1 )
-----
CATG ,A: CHILD CARE
TITL  ,A: ALTERNATIVE TO DR SPOCK
PUBL  ,A: TIME DEC 23 78
INFO  ,A: OTHER WAYS TO KEEP BABY FROM BEDWETTING

( 4 )
-----
CATG ,A: CHILD CARE
TITL  ,A: SUMMER DISHES FOR THOSE IN DIAPERS
PUBL  ,A: GOURMET GUIDE MARCH 79
INFO  ,A: SHOW THEM THAT YOU LOVE THEM
```

This is an appropriate time to discuss a doubt you may have. "This is fine and valuable," you say, "but only after information has been entered into the computer – and just who is going to do that?" The time one spends in organizing information on a personal computer, in the final analysis, will be justified by the usefulness and convenience. It may not be practical, however, for any one person to be in charge of data entry or file management for all members of the family. Therefore, it is recommended that all persons in a position to use such a reference library be trained in the use of common procedures and safeguards. As a precaution against the inadvertent wiping out of the master reference file, each person should make a new entry in a sub-reference file that can be periodically integrated into the master file by the person most familiar with the operation of the microcomputer.

A great deal of useful information that comes across one's casual reading is bypassed for the reason that it can not be conveniently retrieved later when needed. It is hoped that your personal computer with the help of PIMS can provide just the service that is needed to change that.

**A Computer for a Personal Secretary**

Our next application is an unusual one. A personal secretary is expected to answer your telephone, make and keep track of your personal or business appointments, remind you of your appointments, and maybe, present you with a fresh brewed cup of coffee every morning. Even though your computer has the capability of performing all these functions, some (particularly the latter) would require elaborate interface devices. In this example, we shall illustrate how a microcomputer could be used to keep track of your appointments. No special interface required.

The appointment patterns vary with individuals. Some schedules are booked for months in advance, whereas others are booked just a day in advance. The computer's role is to keep track of the following kinds of information:

```
DATE (YR-MO-DY)
TIME OF APPOINTMENT
WHO
PURPOSE
```

An appointment calendar should be readily accessible so that the open spots may be used to oblige late callers. Basically, you would like to be able to know at what times you are booked on a certain day, and if any certain hour is free. The main command used in this application is the SEARCH command. We must choose a code that can be specified when searching to answer the type of questions mentioned above. We shall organize a yearlong appointment file for this illustration – but it can be readily adapted to your needs.

What appointments do you have on a certain day? This question can be answered by simply searching through a DATE field and listing all those records which show the date in question. The code for this can be straightforward. JAN 23 for January 23rd, etc. To be able to search for a particular hour on a given day, both the hour and the day must be combined in a code. Say, 0800 AM FEB 14 for 8 in the morning on February 14th. To enable preparation of a list of the day's appointments plus the ability to search to a specific time and date these codes are entered in separate fields. We chose this ex-

ample to illustrate how, depending on the application, you may have to enter some information more than once. Here is a sample data base:

```
DATE ,A: DEC 24
TIME  ,A: 11 PM DEC 24
WHO   ,A: SANTA CLAUS
WHAT ,A: RECEIVE DELIVERY ON TOYS

DATE ,A: JAN 23
TIME  ,A: 11 AM JAN 23
WHO   ,A: LORI
WHAT ,A: BIG BIRTHDAY PARTY

DATE ,A: DEC 31
TIME  ,A: 12 PM DEC 31
WHO   ,A: OLD YEAR
WHAT ,A: SAY GOOD-BYE

DATE ,A: JAN 23
TIME  ,A: 7 PM JAN 23
WHO   ,A: FIREHOUSE
WHAT ,A: FUNDRAISING DINNER

DATE ,A: JAN 6
TIME  ,A: 8 AM JAN 6
WHO   ,A: DENTIST
WHAT ,A: CAN'T YOU GUESS
```

Now, with an appointment data base established, we wish to be reminded of what has been scheduled for next January 23. Here is the procedure:

```
(COMMAND): ? SEARCH

ENTER FIELD NUMBER TO BE COMPARED? 1
ENTER EXPRESSION TO BE COMPARED? JAN 23
( 2 )
-----
DATE ,A: JAN 23
TIME  ,A: 11 AM JAN 23
WHO   ,A: LORI
WHAT ,A: BIG BIRTHDAY PARTY
( 4 )
-----
DATE ,A: JAN 23
TIME  ,A: 7 PM JAN 23
WHO   ,A: FIREHOUSE
WHAT ,A: FUNDRAISING DINNER
```

Checking to see if a particular day and time is booked can be accomplished by searching the TIME field. Let us say we wish to see if 2 PM on December 31 is free. Two (TIME) is the field number to be compared and 2 PM DEC 31 is the expression to be compared. Anything scheduled for that date and time will appear on the screen. If nothing is scheduled, PIMS will ask for your next command.

**Income Tax Deductions**

In the earlier checkbook application, the only function performed was to get a running balance. That in itself may not justify the investment of time it would take to change your habits to include working on a microcomputer. It is hoped that the present application will offer some added incentive.

The checkbook register contains at least a partial financial record for the majority of those who have a checking account. If you use a check for all transactions, then it contains your total financial records. Other than keeping you from overspending, another way in which this record can be helpful is at tax-filing time. Some people opt to take a standard deduction instead of itemizing simply because of the extra labor involved with the latter. This decision to take the standard deduction is also frequently made with incomplete information. With very little effort, your microcomputer can not only give you the information you need to make the right decision, it can help with virtually all the work required in itemizing your deductions, should you choose to do so.

Along with the fields used to create the data file in the earlier checkbook application, we shall add a TAX field that would classify your disbursements for tax purposes at the time of entry. You may use a numerical code. Say, 1 for those expenses that need to be looked at for tax purposes and 2 for those that don't. Code 1 expenses could be further classified. Designate child care expense by using 1C or medical expense by using 1M and so on. Here is a sample printout:

```
DATE ,A: NOV 12 79
NUM  ,A: C130
DISC ,A: DENTIST
AMT  ,N: 45.00
COMT ,A: CHECKUP
TAX  ,A: 1M
```

At year end you can sum all the deductable expenses to provide the information you need to make a decision on itemization. Follow

this procedure:

1) Sum field 4 (amount) over all records.

2) Sum the amount again, this time entering field 6 (tax) as the field number to be compared and 2 (non-deductable expenses) as the expression to be compared.

3) The difference between these two sums will be the total for your deductable expenses.

This now gives you the key information for deciding whether or not to itemize deductions. Should it be advantageous to itemize, you may sort each classification (medical, child care, etc.) of deductable expense and neatly produce the final list which can be entered directly on your tax form.

**A Tutor for Every Subject**

This example is to illustrate the use of PIMS in tutoring. The application can be adapted to virtually any subject from a fun quiz to serious tutoring in arithmetic. Here is how it works. First you prepare a data file consisting of a question field and an answer field, both alphanumeric. For example:

```
QUIZ  ,A: WHAT IS THE SQUARE ROOT OF 144?
ANS   ,A:12
```

Use of this PIMS application is accomplished with the CHANGE command. Here is the procedure for using PIMS for tutoring once the required data file has been established:

1) Use the CHANGE command and enter A (for all) as the mode.

```
(COMMAND):? CHANGE

ENTER MODE: A(ALL), O(ONE), R(RANGE)
?A
```

2) Read the question and enter your answer following the question mark:

```
( 1 )
-----
QUIZ  ,A: WHAT IS THE SQUARE ROOT OF 144?
? 144
```

3) Depress the enter or return key and the correct answer will be displayed, giving the user instant feedback. Before going on to the

next question, depress the hyphen key to avoid erasing the pre-recorded answer.

When all of the questions have been answered, the user can use the LIST command to inspect his or her answers and compare them with the correct ones. The printout would look like the following:

```
( 1 )
-----
QUIZ  ,A: 143.5
ANS   ,A: 144
( 2 )
-----
QUIZ  ,A: NO
ANS   ,A: YES
( 3 )
-----
QUIZ  ,A: THREE TIMES
ANS   ,A: SEVEN TIMES
```

The user's answer is shown in the first field and the correct answer is found in the answer field.

**How to Give a Test**

An application similar to tutoring is that of administering a test. In this case, however, instead of giving the student instant feedback on the correctness of his or her answer, it is desired that the answer simply be recorded along with the question for later grading. Two fields are used – one for the question, the other for the answer. In preparing the test, the answer field is left blank. A printout would look like this:

```
( 1 )
-----
QUIZ  ,A: WHAT IS YOUR NAME?
ANS   ,A:
( 2 )
-----
QUIZ  ,A: NAME THE SMALLEST STABLE CHARGED PARTICLE.
ANS   ,A:
( 3 )
-----
QUIZ  ,A: WHAT IS PIGSKIN MOSTLY USED FOR?
ANS   ,A:
(COMMAND):?
```

Note that the first question allows for the entry of the student's name. When taking the test, the CHANGE command is used as in the

previous application. It is important with this example, however, that the student respond to the "?" following the test's questions by entering a "—" or hyphen. If it is not entered, it will result in the question being deleted. Here is an example of a proper entry:

```
( 1 )
-----
QUIZ  ,A: WHAT IS YOUR NAME?
?—
ANS   ,A:
? JOHN TESTTAKER
```

At the completion of the test, the student should use the SAVE command to record the test results for later review and scoring. The procedure is illustrated by the following:

```
(COMMAND):? SAVE

ENTER LABEL FOR FILE BEING SAVED? FINAL QUIZ
(COMMAND)?: END

HAVE YOU SAVED ALL FILES PROPERLY?
ENTER Y/N? Y
END OF PERSONAL INFORMATION MANAGEMENT SYSTEM
ANOTHER QUALITY SOFTWARE PRODUCT FROM SCELBI
READY
>
```

When it is time to grade the test or tests, each one may be recalled from the cassette storage as follows:

```
ENTER OPERATION MODE:
1). CREATE NEW FILE 2). LOAD FROM CASSETTE
?2
ENTER FILE NAME? FINAL QUIZ
00000→QUIZ,A→ANS ,A→STOP0→ 3→
 1→WHAT IS YOUR NAME?→JOHN TESTTAKER→
 2→NAME THE SMALLEST STABLE CHARGED PARTICLE.→ELECTRON→
 3→WHAT IS PIGSKIN MOSTLY USED FOR?→TO HOLD PIGS TOGETHER→
EOF
(COMMAND):?
```

Thus, the person administering the test will see first the name of the test taker, then the question followed by the answer to enable scoring of each test.

## Keeping a Log for Your Amateur Radio Station

Keeping a log is a great way of recording a lot of useful data. Retrieving the data can sometimes be difficult. Sorting the log's contents for specific data can be more trouble than it is worth. But not if you have the power of PIMS. This application uses the example of an amateur radio station log. It is directly adaptable to short-wave listening or CB. Other logging-type applications may be devised by suitably structuring the fields. The format of this radio amateur log follows that of SCELBI's *Superlog* as a convenience for those wishing to transcribe entries from written form to a computer. Here is a sample record with data entered:

```
DATE ,A: 79-12-26
TIME ,A: 2200
CALL ,A: WA2ABC
SENT ,N: 599
RCVD ,N: 579
QTH  ,A: CT
BAND ,A: OSCAR 8 2/10
PWR  ,A: 150
QSL  ,A: S
NOTE ,A: HELIX
```

Most of the fields are self-explanatory. The NOTE field can be used for a variety of purposes. In this example it is used to record the particular antenna in use at the time of contact. The SENT and RCVD fields are usually used for recording signal reports. In contest work, however, they may be used for contact number or required exchange info. The QSL field keeps track of whether a QSL card has been sent (S) or received (R).

When your data base contains a large number of records, it becomes useful for quite a variety of things. For example, if you hear K1XYZ transmitting and wish to know if you have worked him before, it is an easy task to check. Use the SEARCH command entering 3 (CALL) as the field number to be compared and K1XYZ as the expression to be compared. The program will search out and display all of the entries, if any, for K1XYZ.

There are countless other chores which can now be performed. Who owes you a QSL card? The answer can be obtained by sorting for all S (sent) entries in field 9 (QSL?). Which antenna is getting you the best signal reports? Enter 5 (RCVD) as the field number to be sorted. Then list the results and observe whether one particular antenna is appearing consistently at the top of the list. Another way of evaluating antennas would be to compare the average signal report

received when using, say, your helix, versus your yagi. This can be done with the SUM command as follows:

```
(COMMAND):? SUM

ENTER FIELD NUMBER TO BE SUMMED? 5
DO YOU WANT TO SUM OVER ALL RECORDS?(Y/N)? N
ENTER FIELD NUMBER TO BE COMPARED? 10
ENTER EXPRESSION TO BE COMPARED? HELIX
( 1 )( 3 )( 5 )( 22 ) SUM OF RCVD ,N = 232
```

The printout indicates that the term "helix" was found in four records and the sum of field 5 for those records is 232. Dividing 232 by 4 indicates an average signal report of 58. Repeating this procedure, but entering "yagi" as the expression to be compared will yield the average signal report with the yagi for comparison. Obviously, when making comparisons such as these, other variables like power, time of day, etc. must be taken in to account, also.

## Chapter 6

# Loading PIMS Into Your Computer

PIMS is written in Microsoft BASIC language. This popular version of BASIC is available on a variety of popular microcomputer systems including those produced by such manufacturers as APPLE, IMSAI, PERTEC (formerly MITS), CROMEMCO, COMMODORE, and RADIO SHACK Corporations. PIMS was originally developed and tested on an ALTAIR 8800b system produced by MITS, Inc. It has also been specifically tested on a Radio Shack TRS-80 level II and a Commodore PET 2001 unit. It is designed to operate (with possibly some minor changes) in virtually any system that utilizes Microsoft BASIC or a compatible BASIC language translater.

If you have a Radio Shack TRS-80 level II with 16K of memory you can load the program exactly as it is shown in this chapter and expect it to operate as described in this publication. If you have a Commodore Pet 2001 with sufficient memory (16K recommended) you can load the program with a few modifications as specifically described in this chapter. If you have any system running Microsoft compatible BASIC, you should be able to implement the program as it is written with a few possible changes required in the bulk storage routines and at a few other points as will be mentioned in this section.

### Loading the Source Listing into Your Computer's Memory

The first step towards implementing PIMS on your own system is, of course, to load the program into your computer's memory. This may be accomplished by typing in the source listing provided in this chapter. Essentially, you should type this program in exactly as it is shown, word for word! The only exceptions would be if you are trying to adapt the program to operate with a different version of BASIC, or if you are an advanced programmer and wish to implement changes to the input/output sections (for the bulk storage device) as you come across those parts.

Admittedly, it is the rare operator that can manually enter a

program the size of PIMS into a computer without making a few undiscovered mistakes. We suggest, to minimize the amount of work required, that you proceed in the following manner: Type in the source listing in short stretches, say an amount that can be keyboarded in a half hour or so. After each session, make a copy of all that you have entered so far on your bulk storage device. This is done as a precaution in the event you have a power failure or some such catastrophe. (Yes. You are specifically authorized to maintain one or more "archival" copies of PIMS for your own use once you have purchased this publication. Remember such copies may not be sold or otherwise distributed, and they must include the copyright notice which is made an integral part of the source listing.) Take a break between keyboarding sessions. Don't get hurried or fatigued. Doing so is likely to result in more keyboarding errors. Proofread each line as it is entered. Do not attempt to skip over "REMark" lines in the listing. A number of routines and subroutines begin with a "REMarks" line. Removing such a line will only result in an error condition when you attempt to run the program. Of course, once you have the entire program entered in memory, you should make a copy of it on your bulk storage device *before* beginning to test it. If necessary correct any typing errors in the program.

How long will it take to enter the program into your computer? It depends on how well you type. Most people with some typing experience will probably find it takes from two to three hours.

**Check Your Work** When you have entered all of the source listing into your system and have made a copy for safekeeping, you are ready to check your work. Perhaps the best suggestion at this point is for you to list out the program (in sections if you are using a video display, or in its entirety if you have a hard copy printer available). Double-check your work against the source listing. Correct any discrepancies that you find.

After performing that procedure, you should be ready to give the program a workout on your machine. If you are running a Radio Shack TRS-80 level II, and you have no clerical errors as a result of the loading procedure, you should be able to begin using PIMS immediately. However, before you start compiling a data base of any value, we suggest you enter a small trial data base and make sure you can utilize all the PIMS commands. You can

try duplicating some of the sample applications given in the preceding chapter. If there are any significant entry errors in your program, your computer will most likely give you a SYNTAX ERROR or other appropriate message along with the offending line number. If that happens, check the designated line to make sure it matches the listing. Make any necessary corrections.

**A Note About Program Line 80**

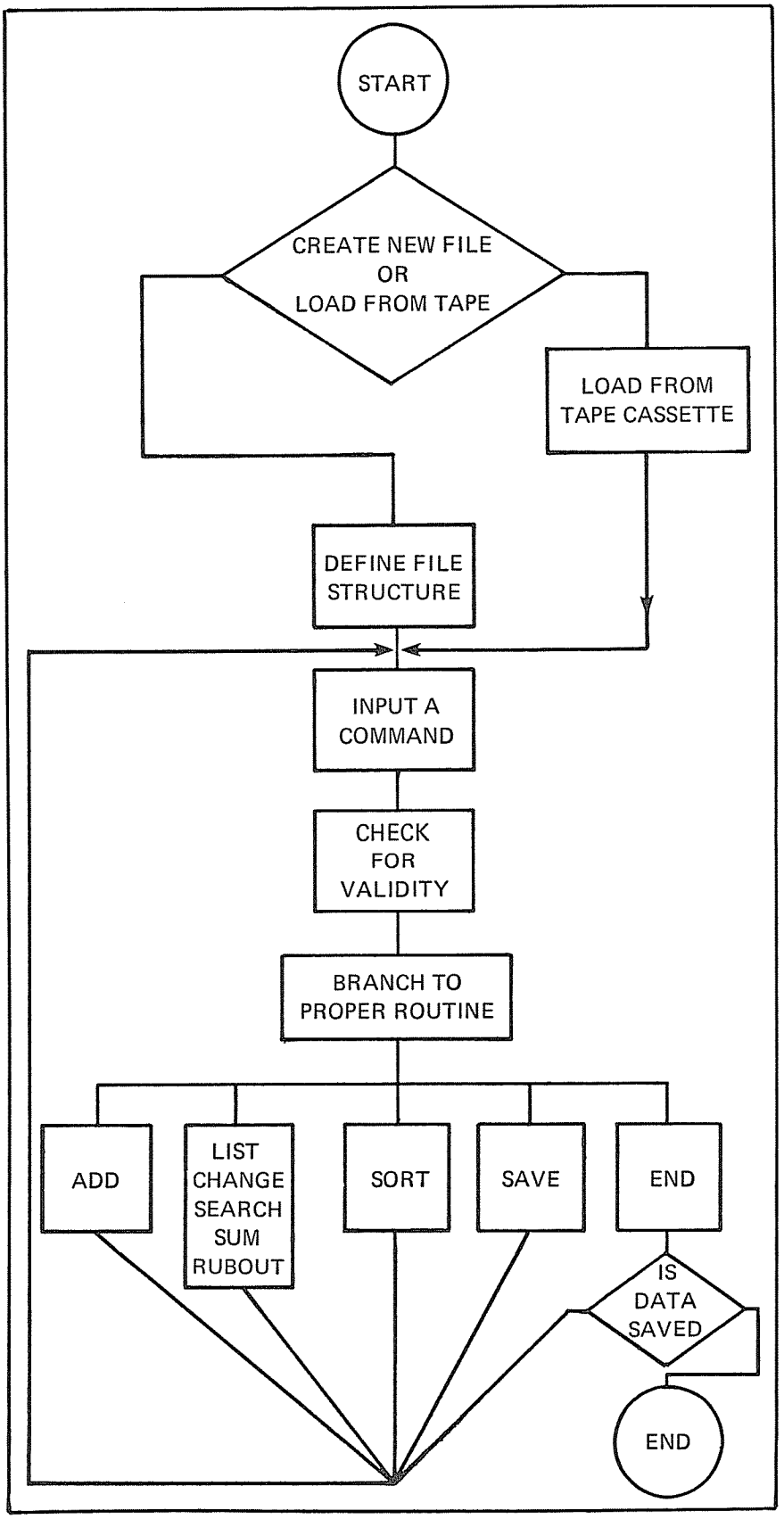
One important note! Line 80 in the program listing as presented for a Radio Shack TRS-80 level II contains a statement that is used to allocate memory space for the user's data file. The statement is shown in the listing as CLEAR 6000, meaning that 6000 bytes of memory are reserved for use by the data base the user will create and manipulate using PIMS. This 6000 byte figure was established based on the assumption that the system had approximately 16K of memory available to the user. If you have more memory available, you can increase the allowable size of the data file by changing the value in the CLEAR statement to a higher number. Thus, if you had a system with 24K of memory, you might set line 80 to read CLEAR 14000. As a final note on this matter, some systems do not need line 80 in the program to reserve strings storage space. Such systems dynamically allocate strings storage space as it is required. In such systems, line 80 may generate an error message when PIMS is first activated. In such cases you can delete the statement from the program.

**Making Alterations**

If you are implementing PIMS on a system that is not running Microsoft compatible BASIC, then you may need to refer to your BASIC instruction manual to substitute appropriate commands or make formatting alterations. Again, if you approach things on a step-by-step basis, you should have little difficulty. The first part of the program (through line number 510) is used to structure a data base. First get that part of the program to work. You can then proceed to debug each type of command if that is necessary. The brief flow chart provided in this chapter may be of some assistance in such cases. However, most such work would consist of simply having your system help you locate the lines in the program that it could not interpret. (Remember, this paragraph refers to the procedures to follow if you are *not* running Microsoft BASIC.)

**Statements and Functions Used by the Program**

As an aid in your determining the fundamental compatability of your BASIC interpreter to that used in creating PIMS, we present


START
CREATE NEW FILE
OR
LOAD FROM TAPE
LOAD FROM
TAPE CASSETTE
DEFINE FILE
STRUCTURE
INPUT A
COMMAND
CHECK
FOR
VALIDITY
BRANCH TO
PROPER ROUTINE
ADD
LIST
CHANGE
SEARCH
SUM
RUBOUT
SORT
SAVE
END
IS
DATA
SAVED
END

here a list of the statement and function types utilized in the program.

| | | |
|---|---|---|
| DATA | DIM | FOR |
| GOSUB | IF – THEN | INPUT |
| NEXT | ON – GOTO | ON – GOSUB |
| PRINT | PRINT USING | READ |
| REM | RETURN | LEFT$ |
| RESTORE | LEN$ | MID$ |
| VAL$ | CHR$ | RIGHT$ |

In addition to the statement types above, there are a few types that are not necessary to the functioning of the program. These have to do with the error trapping capability of some systems such as in the Radio Shack TRS-80 level II model. Lines number 90, 3430 and 3440 refer to these types of statements. If your system does not support these types of statements for error flagging, simply delete those lines from the program.

**Setting Up PIMS on a Commodore PET 2001**

There are only a few areas in PIMS that may require special attention if you are running Microsoft BASIC on a system other than a Radio Shack TRS-80 level II. One such area is that associated with controlling your bulk storage devices. A few other places are shown in the accompanying listing that presents alterations that would be made to run PIMS on a Commodore PET 2001 system.

```
80   REM THIS LINE NOT WANTED FOR A COMMODORE PET 2001
250  PRINT I;
280  IF T$="STOP" GOTO 420

1150  PRINT "LOAD & ALIGN LABELS; HIT RETURN"
1155  GET T$
1157  IF T$="" GOTO 1155

1320  PRINT "ENTER EXPRESSION TO BE COMPARED"
1325  INPUT S$
```

Line 80 is not needed in a Commodore PET 2001 system because the unit dynamically allocates string storage space as it is required. Leaving line 80 in will generate an error message. Line 280 is changed so that the program looks for the word STOP to conclude the data file structuring routine, instead of an empty field, since the system does not like to accept a blank line as an input. The remaining alterations shown take care of minor discrepancies between the manner in which inputs are accepted on the Commodore

Pet 2001 versus the Radio Shack TRS-80 level II systems. You might have to make similar alterations in those areas depending on your system's capabilities.

Modifying PIMS to properly control the Commodore PET's tape cassette unit requires changes in the sections related to the SAVE command, as well as the section that is used to load in a tape containing a file of data. The two listings below illustrate the suggested changes specifically for a 2001 system and indicate the general areas in which changes would typically be made for other types of systems. Refer to your system's user's manual for details on how to record and recover data using your bulk storage device. Make any necessary changes to the program in the areas indicated by these examples.

```
1890 '------------------- *LOAD COMMODORE PET ROUTINE
1900 CLOSE 1
1910 J= - 1
1920 INPUT "ENTER FILE NAME";F$
1930 OPEN 1,1,0,F$
1940 J=J + 1
1950 INPUT#1,T$
1960 IF (ST) AND (64) GOTO 2000
1970 PRINT T$
1980 R$(J)=T$
1990 GOTO 1940
2000 T$=R$(0)
2010 T1$=CHR$(126)
2020 GOSUB 3040
2030 FOR I=0 TO 10
2040 N$(I)=B$(I)
2050 IF LEFT$(N$(I),4)="STOP" GOTO 2070
2060 NEXT I
2070 N=VAL(B$(I + 1))
2075 CLOSE 1
2080 GOTO 540

3180 '------------------- *SAVE COMMODORE PET ROUTINE
3190 CLOSE 1
3200 INPUT "ENTER LABEL FOR FILE BEING SAVED";F$
3204 POKE 243,122
3208 POKE 244,2
3210 OPEN 1,1,1,F$
3220 T$="00000"
3230 FOR I=1 TO 10
3240 T$=T$ + CHR$(126) + N$(I)
3250 T1$=LEFT$(N$(I),4)
3260 IF T1$="STOP" THEN 3280
3270 NEXT I
```

```
3280  T$=T$ + CHR$(126) + STR$(N) + CHR$(126)
3290  PRINT#1, CHR$(34) + T$ + CHR$(34)
3300  FOR J=1 TO N
3310  PRINT#1, CHR$(34) + R$(J) + CHR$(34)
3320  NEXT J
3325  CLOSE 1
3330  RETURN
```

*PET Enhancement Routine*

The Commodore PET has a limitation in its handling of data strings which normally limits such to about 70 characters. If your records contain more than this number of characters, an enhancement routine like the one shown on page 82 may be required.

**The Source Listing**

After you have worked with PIMS for awhile, you might find that you want to customize the program by adding some of your very own special features. Since we have provided the full source listing in this publication, you will be in a position to make whatever changes you desire. There are ten line numbers available between most statements in the program so you have plenty of room to add statements between line numbers. Experiment – PIMS is now *your* program!

```
10  PRINT "*PIMS – PERSONAL INFORMATION MANAGEMENT SYSTEM*"
20  PRINT "*COPYRIGHT (C) 1979 BY SCELBI C. C., INC."
30  PRINT "-----------------------------------------------------"
40  PRINT
50  REM WRITTEN BY: MADAN L. GUPTA
60  REM COPYRIGHT (C) 1979 BY SCELBI C. C., INC."
70  REM INITIALIZED FOR UP TO 10 FIELDS IN UP TO 99 RECORDS"
80  CLEAR 6000     '*****USER MAY NEED TO MODIFY THIS*****
90  ON ERROR GOTO 3430
100  DIM R$(100), D$(100)
110  DIM N$(10), B$(20), B(20)
120  PRINT "ENTER OPERATION MODE:"
130  PRINT "1). CREATE NEW FILE  2). LOAD FROM CASSETTE"
140  WM%=0
150  INPUT WM%
160  ON WM% GOTO 170,1890
170  N=0    'NUMBER OF RECORDS IN FILE
180  B$=""
190  B1$="      "    '5 SPACES IN HERE
200  PRINT "DEFINE FILE STRUCTURE; ENTER FIELD NAME,TYPE(A OR N)"
210  N$(0)="EMPTY"
220  FOR I=1 TO 10
230  T$=""
240  T1$=""
250  PRINT USING "##"; I;
260  PRINT ".  ";
```

```
270  INPUT T$,T1$
280  IF T$="" THEN GOTO 420
290  IF T1$<>"A" THEN GOTO 310
300  GOTO 350
310  IF T1$<>"N" THEN GOTO 330
320  GOTO 350
330  PRINT "ILLEGAL TYPE ENTRY; ENTER AGAIN"
340  GOTO 270
350  T$=T$+B1$
360  T$=LEFT$(T$,4)
370  T$=T$ + "," + T1$
380  GOSUB 2470    'REMOVE SURROUNDING SPACES
390  N$(I)=T$
400  NEXT I
410  GOTO 430
420  N$(I)="STOP0"    '5'TH CHARACTER IN STRING IS A ZERO
430  PRINT "STRUCTURE DEFINITION COMPLETE"
440  PRINT "FIELDS ARE:"
450  PRINT
460  FOR I=1 TO 10
470  IF LEFT$(N$(I),4)="STOP" THEN GOTO 500
480  PRINT I;" : ";N$(I)
490  NEXT I
500  PRINT
510  GOTO 540
520  '-------------------
530  PRINT "ILLEGAL COMMAND; ENTER AGAIN"
540  INPUT "(COMMAND):";T$
550  N1=0    'TRANSACTION NUMBER
560  PRINT
570  RESTORE
580  READ Z$,T
590  IF Z$="##" THEN 530
600  IF LEFT$(Z$,3)<>LEFT$(T$,3) THEN 580
610  ON T GOSUB 750,1120,1120,1120,2090,3340,1310,1220,3180,1120,2620
620  GOTO 540
630  DATA "ADD",1
640  DATA "LIST",2
650  DATA "CHANGE",3
660  DATA "LABELS",4
670  DATA "HELP",5
680  DATA "END",6
690  DATA "SEARCH",7
700  DATA "SUM",8
710  DATA "SAVE",9
720  DATA "RUBOUT",10
730  DATA "SORT",11
740  DATA "##",-1
750  '-------------------    *ROUTINE TO ADD RECORDS TO FILE
760  N=N+1
770  N1=N1 + 1
```

```
780  T$=""
790  PRINT "(";N;")"
800  PRINT "- - - - -"
810  FOR I=1 TO 10
820  IF LEFT$(N$(I),4)="STOP" THEN GOTO 980
830  PRINT N$(I);": ";
840  T1$=" "     '1 SPACE IN HERE
850  INPUT T1$
860  IF T1$="" THEN T1$=" "     '1 SPACE IN SECOND STRING
870  IF T1$="STOP" THEN 1060
880  IF MID$(N$(I),6,1)<>"N" THEN 920
890  IF ASC(T1$)>43 AND ASC(T1$)<58 THEN 920
900  PRINT "**ENTER NUMERIC INFORMATION ONLY**"
910  GOTO 850
920  REM STRING OK
930  REM SO ADD STRING
940  REM TO RECORD
950  IF LEN(T$) + LEN(T1$)>245 GOTO 1010
960  T$=T$ + CHR$(126) + T1$
970  NEXT I
980  N$=STR$(N)
990  T$=N$ + T$ + CHR$(126)
1000  GOTO 1030
1010  PRINT "RECORD LENGTH EXCEEDED; RE-ENTER RECORD"
1020  GOTO 780
1030  R$(N)=T$
1040  PRINT
1050  GOTO 760
1060  PRINT "END OF ADDITION"
1070  N=N - 1
1080  N1=N1 - 1
1090  PRINT N1; " RECORDS ADDED"
1100  RETURN
1110  '- - - - - - - - - - - - - - - - - - -
1120  GOSUB 2170    'GET RANGE OF RECORDS TO PROCESS
1125  IF T=10 THEN 1465
1130  IF T<>4 THEN 1360
1140  FOR I=1 TO 3
1150  INPUT "LOAD & ALIGN LABELS; HIT RETURN";W9%
1160  FOR J=1 TO 5
1170  PRINT "XXXXXXXXXXXXXXXXXXXXXXXXX"
1180  NEXT J
1190  PRINT
1200  NEXT I
1210  GOTO 1360
1220  '- - - - - - - - - - - - - - - - - -    *SEARCH & SUM ROUTINE
1230  INPUT "ENTER FIELD NUMBER TO BE SUMMED"; SM%
1240  IF MID$(N$(SM%),6,1)="N" THEN 1270
1250  PRINT "IMPROPER FIELD TYPE; ENTER AGAIN"
1260  GOTO 1230
1270  INPUT "DO YOU WANT TO SUM OVER ALL RECORDS?(Y/N)";S1$
```

```
1280  IF S1$<>"Y" THEN 1310
1290  S%=0
1300  GOTO 1330
1310  INPUT "ENTER FIELD NO. TO BE COMPARED";S%
1320  INPUT "ENTER EXPRESSION TO BE COMPARED";S$
1330  T1=1
1340  T2=N
1350  SM=0
1360  FOR I=T1 TO T2
1370  T1$=R$(I)
1410  T$=LEFT$(T1$,5)
1420  T3=VAL(T$)
1430  T$=T1$
1440  T1$=CHR$(126)
1450  GOSUB 3050    'PARSE STRING
1460  GOTO 1490
1465  FD=T2 - T1 + 1    'RUBOUT COMMAND
1470  FOR I=T2 + 1 TO N + 1
1475  R$(I - FD)=R$(I)
1480  NEXT I
1484  N=N - FD
1488  RETURN
1490  IF T<=4 THEN 1570
1500  IF S%=0 THEN 1520    'SUM ALL RECORDS
1510  IF B$(S%)<>S$ THEN 1850    'SEARCH & SUM ROUTINE
1520  IF T=7 THEN 1650
1530  PRINT "(";I;")";
1540  T3=VAL(B$(SM%))
1550  SM=SM + T3
1560  GOTO 1850
1570  ON T - 1 GOTO 1650,1650,1580
1580  '------------------    PRINT LABELS ROUTINE
1590  FOR J=1 TO 5
1600  PRINT B$(J)
1610  NEXT J
1620  PRINT
1630  GOTO 1850
1640  '------------------
1645  PRINT "RECORD LENGTH EXCEEDED; RE-ENTER RECORD"
1650  PRINT "(";I;")"
1660  PRINT "-----"
1670  FOR J=1 TO 10
1680  IF LEFT$(N$(J),4)="STOP" THEN GOTO 1760
1690  PRINT N$(J);":  ";B$(J)
1700  IF T<>3 THEN 1740
1710  INPUT T1$
1720  IF T1$="—" THEN GOTO 1740
1730  B$(J)=T1$
1740  NEXT J
1750  '------------------
1760  IF T<>3 THEN 1850
```

```
1770  N$=B$(0)    'ADD CHANGED RECORD HERE
1780  PRINT N$
1790  T$=N$ + CHR$(126)
1800  FOR J=1 TO 10
1805  IF LEN(T$) + LEN(B$(J))>245 GOTO 1645
1810  T$=T$ + B$(J) + CHR$(126)
1820  IF N$(J)="STOP0" THEN 1840
1830  NEXT J
1840  R$(I)=T$
1850  NEXT I
1860  IF T<>8 THEN 1880
1870  PRINT "SUM OF  ";N$(SM%);" = ";SM
1880  RETURN
1890  '-------------------    *LOAD FROM CASSETTE
1900  REM
1910  INPUT "ENTER FILE NAME";F$
1914  INPUT# - 1, T$
1918  IF T$=F$ THEN GOTO 1950
1922  PRINT "FOUND FILE:  ";T$
1926  INPUT "CONTINUE SEARCH (Y/N)  "; T$
1930  IF T$="N" THEN 540
1934  INPUT# - 1, T$
1938  IF T$<>"EOF" THEN 1934
1942  GOTO 1914
1946  J=J + 1
1950  INPUT# - 1, T$
1960  PRINT T$
1965  IF T$="EOF" THEN 1990
1970  R$(J)=T$
1980  GOTO 1946
1990  T$=R$(0)
2000  T1$=CHR$(126)
2010  GOSUB 3040
2020  FOR I=0 TO 10
2030  N$(I)=B$(I)
2040  IF LEFT$(N$(I),4)="STOP" THEN 2060
2050  NEXT I
2055  I=I-1
2060  N=VAL(B$(I + 1))
2070  GOTO 540
2090  '-------------------    *HELP COMMAND
2100  PRINT "VALID COMMANDS ARE"
2110  RESTORE
2120  READ Z$,T
2130  IF Z$="##" THEN 2160
2140  PRINT Z$
2150  GOTO 2120
2160  RETURN
2170  '-------------------    *GET RANGE
2180  IF N=0 THEN PRINT "FILE IS EMPTY"
2190  IF N=0 THEN 540
```

```
2200  PRINT "ENTER MODE: A(ALL), O(ONE), R(RANGE)"
2210  INPUT T$
2220  IF T$<>"A" THEN 2260
2230  T1=1
2240  T2=N
2250  GOTO 2430
2260  IF T$<>"O" THEN 2310
2270  INPUT "WHICH ONE";T1
2280  IF T1>N THEN 2440
2290  T2=T1
2300  GOTO 2430
2310  IF T$="R" THEN 2340
2320  PRINT "ILLEGAL ENTRY; ENTER AGAIN"
2330  GOTO 2200
2340  PRINT "ENTER LOWER BOUND"
2350  INPUT T1
2360  IF T1<1 THEN T1=1
2370  IF T1>N THEN GOTO 2440
2380  INPUT "ENTER UPPER BOUND";T2
2390  IF T2>N THEN 2440
2400  IF T2=>T1 THEN 2430
2410  PRINT "ERROR: UPPER BOUND < LOWER BOUND; ENTER AGAIN"
2420  GOTO 2340
2430  RETURN
2440  PRINT "THIS FILE ONLY HAS ";N;" RECORDS"
2450  GOTO 2200
2460  '--------------------
2470  REM REMOVE SURROUNDING BLANKS FROM A STRING
2480  B$=" "    '1 SPACE HERE
2490  T1=LEN(T$)
2500  IF T1=0 THEN 2600
2510  FOR T2=2 TO T1
2520  IF MID$(T$,T2 - 1,1)<>B$ THEN 2550    'LEADING SPACES
2530  T$=MID$(T$,2,T1 - 1)
2540  NEXT T2
2550  T1=LEN(T$)
2560  IF T1=0 THEN 2600
2570  IF RIGHT$(T$,1)<>B$ THEN 2600    'TRAILING SPACES
2580  T$=LEFT$(T$,T1 - 1)
2590  GOTO 2570
2600  RETURN
2610  PRINT "*** RE-ENTER ***"
2620  INPUT "ENTER FIELD NUMBER TO BE SORTED";S%
2630  PRINT "FIELD FOR SORT IS:  ";N$(S%)
2640  INPUT "DO YOU WANT TO CHANGE IT? (Y/N)";S$
2650  IF S$="Y" THEN 2610
2660  T1=1
2670  T2=N
2680  PRINT "*** EXTRACTING KEY FIELD ***"
2690  FOR I=T1 TO T2
2700  T$=R$(I)
```

```
2710  T1$=CHR$(126)
2720  GOSUB 3040
2730  D$(I)=B$(S%)
2740  NEXT I
2750  IF MID$(N$(S%),6,1)<>"A" THEN 2780
2760  SR%=1
2770  GOTO 2820
2780  IF MID$(N$(S%),6,1)="N" THEN 2810
2790  PRINT "DATA ENTRY ERROR; IMPROPER FIELD TYPE; RUN ABORTED"
2800  RETURN
2810  SR%=2
2820  PRINT "*** START OF SORT ***"
2830  M=N
2840  M=INT(M/2)
2850  IF M=0 THEN 3020
2860  J=1
2870  K=N - M
2880  I=J
2890  L=I + M
2900  IF SR%=2 THEN 2930
2910  IF D$(I)<D$(L) THEN 2990
2920  GOTO 2940
2930  IF VAL(D$(I))<VAL(D$(L)) THEN 2990
2940  T$=D$(I)
2942  D$(I)=D$(L)
2944  D$(L)=T$
2950  T$=R$(I)
2952  R$(I)=R$(L)
2954  R$(L)=T$
2960  I=I - M
2970  IF I<1 THEN 2990
2980  GOTO 2890
2990  J=J + 1
3000  IF J>K THEN 2840
3010  GOTO 2880
3020  PRINT "*** END OF SORT ***"
3030  RETURN
3040  '-------------------
3050  REM ROUTINE TO PARSE STRING
3060  K = - 1
3070  FOR J2=1 TO LEN(T$)
3080  IF T1$=MID$(T$,J2,1) THEN 3110
3090  NEXT J2
3100  RETURN
3110  M%=J2 - 1
3120  K=K + 1
3130  B$(K)=""
3140  B$(K)=MID$(T$,1,M%)
3150  T$=MID$(T$,J2 + 1)
3160  J2=0
3170  GOTO 3070
```

```
3180 '------------------- *SAVE COMMAND
3190 INPUT "ENTER LABEL FOR FILE BEING SAVED";F$
3200 PRINT# - 1, CHR$(34) + F$ + CHR$(34)
3210 T$="00000"
3220 FOR I=1 TO 10
3230 T$=T$ + CHR$(126) + N$(I)
3240 T1$=LEFT$(N$(I),4)
3250 IF T1$="STOP" THEN 3270
3260 NEXT I
3270 T$=T$ + CHR$(126) + STR$(N) + CHR$(126)
3280 PRINT# - 1, CHR$(34) + T$ + CHR$(34)
3290 FOR J=1 TO N
3300 PRINT# - 1, CHR$(34) + R$(J) + CHR$(34)
3310 NEXT J
3320 PRINT# - 1, "EOF"
3330 RETURN
3340 '------------------- *END PROGRAM
3350 PRINT "HAVE YOU SAVED ALL FILES PROPERLY?"
3360 INPUT "ENTER Y/N";A1$
3370 A1$=LEFT$(A1$,1)
3380 IF A1$<>"Y" THEN 540
3400 PRINT "END OF PERSONAL INFORMATION MANAGEMENT SYSTEM"
3410 PRINT "ANOTHER QUALITY SOFTWARE PRODUCT FROM SCELBI"
3420 END
3430 PRINT "ERROR ";ERR;" IN LINE ";ERL
3440 RESUME 540
```

**If you are using a line printer, enter the following changes:**

```
1170 LPRINT "XXXXXXXXXXXXXXXXXXXXXXXXX"
1190 LPRINT " "
1600 LPRINT " ";B$(J)
1620 LPRINT " "
```

**PET Enhancement Routine — see page 75**

```
1915 C$=CHR$(197)+CHR$(207)+CHR$(204)
1945 R$(J)=""
1964 IF T$=C$ THEN 1980
1968 R$(J)=R$(J)+T$
1970 GOTO 1950
1980 PRINT R$(J)
3295 PRINT#1,CHR$(197)+CHR$(207)+CHR$(204)
3301 F=0
3302 O$=LEFT$(R$(J),70)
3303 J1=LEN(R$(J))-70
3304 IF J1<=0 THEN 3308
3306 R$(J)=RIGHT$(R$(J),J1)
3308 PRINT#1,CHR$(34)+O$+CHR$(34)
3309 IF F=1 THEN 3312
3310 IF LEN(R$(J))-70<=0 THEN F=1
3311 GOTO3302
3312 PRINT#1,CHR$(197)+CHR$(207)+CHR$(204)
```

# Index

# Bibliography

Here is a bibliography of some SCELBI books which should be of interest to the user of PIMS. A complete catalog of publications and products is available upon request. Write to: SCELBI Publications, PO Box 133 PP STN, Milford, CT 06460.

SCELBI BYTE PRIMER. The ultimate handbook about microcomputers. Over 400 pages of solid, basic information. Decribes what can be done with a microcomputer. How microcomputers work is treated thoroughly. Introduces the 6800, 8080 and Z80 CPU chip capabilities. All about building your own microcomputer system. Programming for the beginner. Plus much more.

LEARN MICRO-COMPUTERS. A new multimedia information package. Includes text (*Understanding Microcomputers*) plus high-quality cassette. For the beginner just starting in microcomputers. All the fundamentals behind the operation of virtually every microcomputer. Tells what to look for in buying a microcomputer. Companion tape includes chapter-by-chapter synopsis of the book. Key review questions. References to page numbers for further review. A great new idea for self study.

TAKE MY COMPUTER . . . PLEASE! An uproariously funny full-length book about the true-to-life midadventures of author Steve Ciarcia and his computer's inability to cooperate. Page after page of computerized jollies and even illustrations.

MICROCOMPUTER POTPOURRI. Data at your fingertips. A pocket-sized reference for the beginner . . . for the technician . . . for the engineer. Contains a really great glossary that covers all the jargon. Special section reviews all the popular microprocessor chips in detail. Includes a complete text on understanding microcomputers in digest form. It's the handiest microcomputer guide yet.

CALCULATING WITH BASIC. A variety of programs in BASIC language to help the student, scientist, engineer, technician or hobbyist apply the language to practical problems. Covers mathematics, finance and statistics, mechanical engineering and electronics. For fun between such serious applications, Hangman and Space Capture games are also provided.

## PIMS Save and Load Routines Update for PET 2001 Microcomputer

```
1890 REM--------------------
1900 REM
1910 INPUT "ENTER FILE NAME"; F$
1915 OPEN 1,1,0,F$
1935 FOR I-1 TO 10
1940 INPUT#1,N$(I)
1945 IF LEFT$(N$(1),4)="STOP" THEN 1955
1950 NEXT I
1955 I1=I-1
1960 INPUT#1,T$
1965 N=VAL(T$)
1970 FOR J=1 TO N
1975 R$(J)=STR$(J)+CHR$(126)
1980 FOR K=1 TO I1
1985 INPUT#1,T$
1990 R$(J)=R$(J)+T$+CHR$(126)
1995 NEXT K,J
2000 CLOSE 1
2070 GOTO 540
3180 REM--------------------------------
3190 INPUT "ENTER LABEL FOR FILE BEING SAVED";F$
3195 POKE 243,122:POKE 244,2
3200 OPEN 1,1,1,F$
3220 FOR I=1 TO 10
3230 PRINT#1,CHR$(34)+N$(I)+CHR$(34)
3235 POKE 59411,53
3240 T1$=LEFT$(N$(I),4)
3250 IF T1$="STOP" THEN 3262
3260 NEXT I
3262 F=I-1
3270 PRINT#1,CHR$(34)+STR$(N)+CHR$(34)
3285 POKE 59411,53
3290 FOR J=1 TO N
3292 T$=R$(J)
3294 T1$=CHR$(126)
3296 GOSUB 3040
3298 FOR K=1 TO F
3300 PRINT#1,CHR$(34)+B$(K)+CHR$(34)
3301 POKE 59411,53
3302 NEXT K
3310 NEXT J
3325 CLOSE 1
3330 RETURN

2800 RETURN
3030 RETURN
```

*NOTE:* Lines 3195, 3235, 3285, 3301 are required for older PET 2001 units. See users manual for details.

Make no mistake about it. PIMS is indeed a data base management program. You can use it as an alternative to programming. Just define the job you want a microcomputer to do. Express yourself in simple commands and statements. Then the microcomputer plus PIMS does the rest.

PIMS is carefully customized for the small system owner. It is prepared using Microsoft-compatible BASIC as used in a wide variety of personal computers. Systems like the Apple II, Radio Shack TRS-80 level II, Commodore PET, Ohio Scientific and many other small computer systems.

You can define and construct your own data bases. Each record can contain up to ten fields. You can define what goes in each. Then you can search, list and sort. There is even a command that lets you sum columns of numbers. Once established, you can save your data base for later recall on your cassette bulk storage unit. Your data base can be kept current using the simple add, rubout and change commands. Complete source listing is included. PIMS is ready to use on your computer!