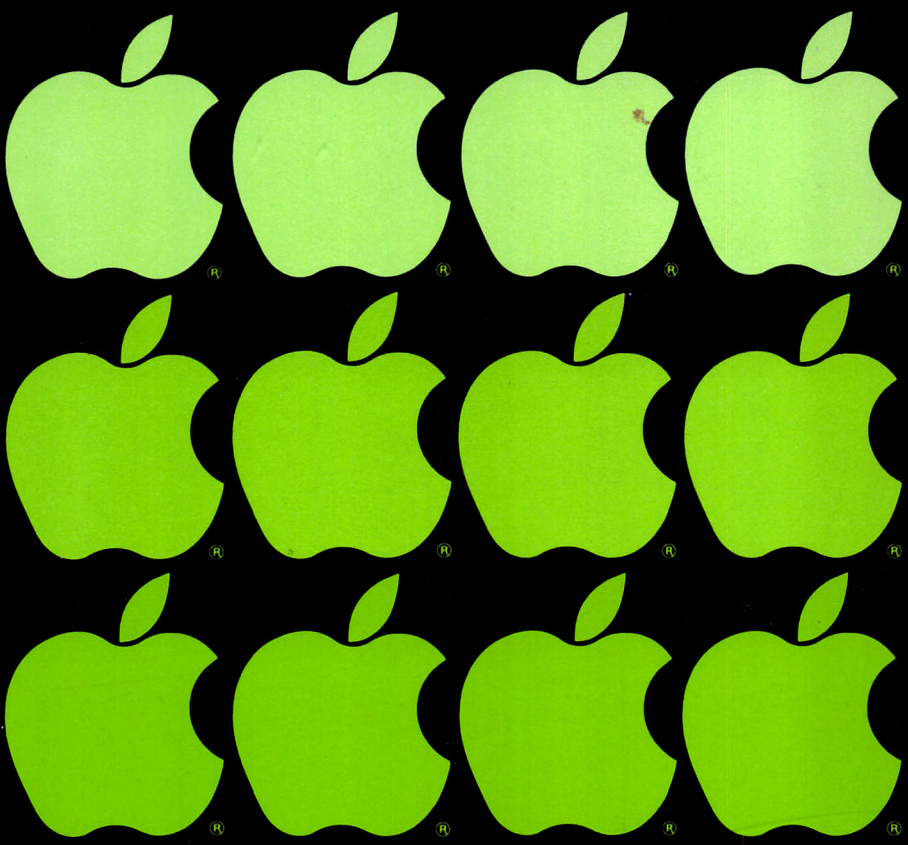
Shake hands with the Apple //e



Pam Kelly-Hartley Joy McKneil

Pitman

Shake hands with the Apple//e

Pam Kelly-Hartley
TDipT, TDipPS, DipTeaching
Technical Teacher
Seven Hills College of Technical and Further Education
Queensland

Joy McKneil
GradDipEdAdmin, DipBusStudies,
DipTeaching, TDipT, DipIPSA
Senior Instructor Business and General Studies
South Brisbane College of Technical and Further Education
Queensland

Pitman

First published 1984

Pitman Publishing Pty Ltd
(Incorporated in Victoria)
158 Bouverie Street
Carlton
Victoria 3053

Level 12
Town Hall House
452-462 Kent Street
Sydney
New South Wales 2000

9th Floor
National Bank Building
420 George Street
Brisbane
Queensland 4000

National Library of Australia
Cataloguing in Publication data

McKneil, Joy, 1935 –
Shake hands with the Apple//e.

Includes index.
ISBN 0 85896 044 3.

1. Apple II (Computer). I. Kelly-Hartley, Pam, 1948 – . II. Title.

001.64'04

Designed by
Sue Veitch

Printed in Australia
by Globe Press Pty Ltd

Associated companies

Pitman Publishing Ltd
London

Copp Clark Pitman
Toronto

Pitman Learning Inc
Belmont, California

Pitman Publishing New Zealand Ltd
Wellington

Contents

18 Word processing 110

Preface

We have been teaching computer operation to post-secondary students for a number of years, and have found that all the manuals currently available assume that the users have a fair understanding of computing terminology.

Shake hands with the Apple//e is a simplified guide for the beginner who has little or no knowledge of computers or computing terms. We have enjoyed writing it and we hope that you will enjoy learning to operate a computer.

Pam Kelly-Hartley and Joy McKneil

Acknowledgments

We wish to thank Apple Computer Australia Pty Ltd for providing photographs and also express our appreciation to those who have given us encouragement, advice and guidance.

1 Introduction

Shake hands with the Apple//e has been designed as an individualised programme to help you learn the operation of the Apple//e microcomputer.

All operating instructions relate specifically to the Apple//e, but in most cases they will also apply to other versions of Apple microcomputers, but may not apply to other brands of microcomputers. However, the knowledge of Apple//e operation will help you to understand how other microcomputers operate.

Figure 1 The keyboard, monitor and disk drive of an Apple //e microcomputer.

All computer systems consist of:

- **hardware:** the physical components of the computer
- **software:** the programs which enable the hardware to be used
- **firmware:** the programs permanently stored in read-only memory.

Brief explanation of terms

Inside the Apple//e

Inside the Apple//e are a number of parts which enable the microcomputer to operate. Some of those parts are:

6502 processor

- the Central Processing Unit (CPU)
- contains special machine-language programs, which enable the CPU to collect information, follow instructions and produce results.

Memory

Computer memory is measured in **bytes.** One byte can contain one character (or similar amount of data).

There are 1024 **bytes** in one **kilobyte** and the capacity of computer memory is normally expressed in kilobytes (K),

eg 16 K of memory = 1024 × 16 = 16 384 bytes
64 K of memory = 1024 × 64 = 65 536 bytes

Read-only memory (ROM) — contains 16 kilobytes of permanent programs to understand and respond to the instructions given by the operator. Read-only memory contains the BASIC language interpreter. This memory is constant, ie it never changes.

Random-access memory (RAM) — data and/or instructions entered at the keyboard or obtained from diskette are temporarily retained in RAM to enable the current task to be performed. This memory is not constant, ie it retains data and/or instructions until these are replaced or until the computer is switched off. All details stored in RAM are lost when the power supply to the computer is turned off.

The Apple//e contains 64 kilobytes of random-access memory.

Circuit boards with integrated circuits

a the main circuit board
b special circuit boards.

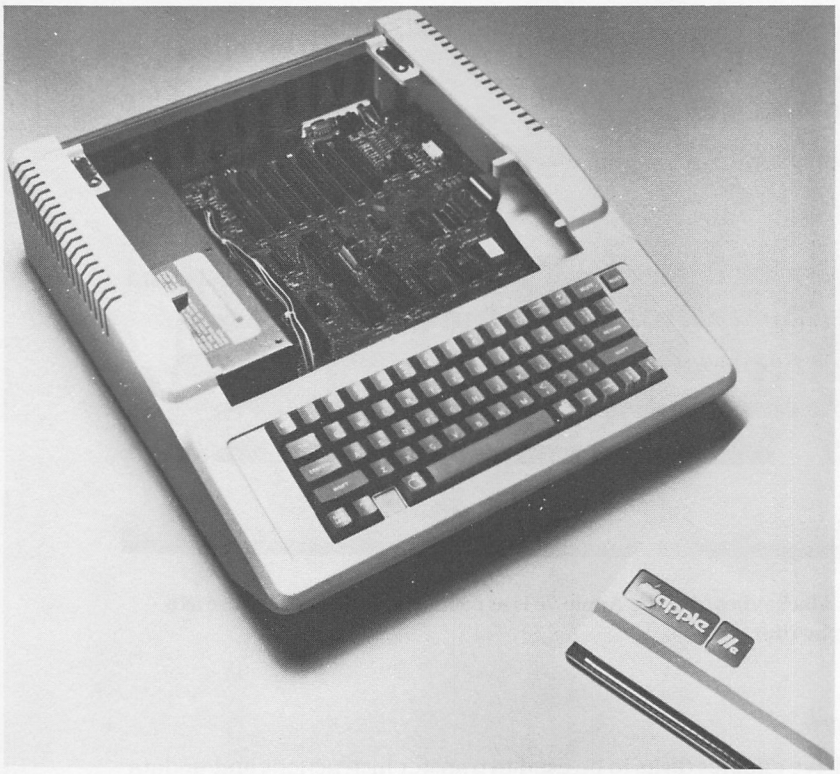


Figure 2 Inside an Apple //e, showing the power supply, main circuit board and slots for special cards which extend the RAM or ROM memory or connect your Apple to a printer or other input/output device.

Display screen

The screen will display:

- text, ie capital letters A to Z, lower case letters a to z, figures 0 to 9 and special symbols
- graphics characters
- special characters.

The screen display allows the operator to see data and instructions entered through the keyboard or obtained from diskettes, and the results of those instructions.

Keyboard

You will communicate with the Apple//e by using a keyboard very similar to a typewriter keyboard.

The keyboard is displayed in Figure 3. Study it carefully and refer to it when new keys are introduced.

Figure 3 The keyboard of the Apple //e is as comfortable and easy to use as a standard typewriter.

Disk drive

The disk drive enables disks to be used to transfer instructions and/or data to and from RAM and to provide a means of storing programs developed on the computer.

The operation of the disk drive can be compared to that of a cassette recorder. When the disk drive door is closed (similar to depressing the 'Play' button on a cassette recorder) the 'head' is lowered towards the disk to read the contents.

Speaker

An inbuilt speaker enables the Apple//e to produce sounds, which include a beep as a signal to the operator, and musical tones.

Disks

A disk (or diskette) is a circular vinyl disk protectively enclosed in a flexible plastic envelope. A slot in the envelope allows access for the head of the disk drive to read the contents of the disk, while a notch on the side of the envelope will determine whether information may be written to the disk. If this notch is covered, the disk is 'write-protected', ie new information cannot be recorded and so the information already on the disk will not be destroyed.

Cassette recorder

You may also use a good quality cassette recorder to transfer programs and data on cassette tape to and from RAM.

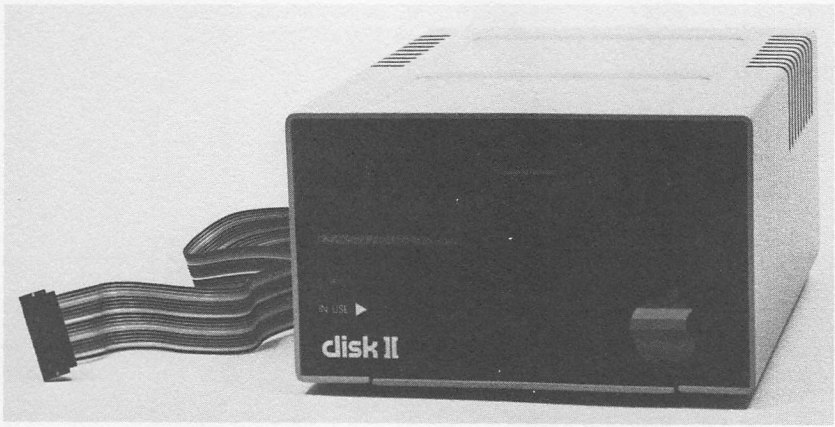


Figure 4 The disk drive of an Apple //e.

Cassettes

Audio cassettes may be used for storage of programs, for loading pre-recorded programs into the computer and for storing your own programs.

Printer

The printer is used to produce a printed copy of output from the microcomputer.
There are a number of printers which may be connected to the Apple//e.
The correct size and type of paper must be available for the type of printer used.

Languages

The Apple//e can operate with a number of computer languages. One of the easiest languages for beginners to learn is known as BASIC.

BASIC is an acronym for:

Beginners **A**ll-purpose **S**ymbolic **I**nstruction **C**ode

The Apple//e supports two versions of BASIC. These are:

- Applesoft BASIC — permanently stored in ROM.
- Integer BASIC — loaded from a master diskette.

All instructions in this book relate to Applesoft BASIC language.

2 First handshake

In this section you do not use disks or cassettes.

You will be working under the directions of the programs permanently stored in ROM and all data and/or instructions entered at the keyboard will be temporarily stored in RAM.

Starting the system

1 Turn the system on at the power switch located at the left rear of the machine, next to the power cord inlet.

2 The system will beep and the green power light (bottom left of keyboard) should now be ON.

Title APPLE][will appear at the top of the screen.

3 If a disk drive unit is connected:

a The red IN USE light on the disk drive unit will light up and the motor will start to whirr. As no disk is inserted it is on an 'endless search'. This basically means that the head in the disk drive is trying to read what is on the disk, but as no disk is inserted it will continue this fruitless search.

b Press CONTROL and RESET keys together.

4 The screen will now display two symbols: a prompt (]) and a flashing cursor (■).

The prompt indicates the type of BASIC language that the computer will read.

Prompt] = Applesoft BASIC

The cursor indicates the typing position on the screen.

A quick look at the keyboard

The keyboard is very similar to the keyboard of an ordinary typewriter, but there are important differences.

Touch

Touch will be slightly different from that of an electric typewriter and totally different from a manual typewriter's.

SHIFT keys

There are two SHIFT keys, one on each side of the keyboard. Depress either SHIFT key in conjunction with another key to produce either:

- an uppercase letter, eg A, Z or
- the character printed on the top portion of the key, eg ! $?

Examples

1 Depress and hold SHIFT; press A. These two actions are abbreviated as SHIFT – A, and produce uppercase A.

2 SHIFT – 4 will produce the $ symbol.

CAPS LOCK

CAPS LOCK is located at the bottom left of the keyboard.

Depress CAPS LOCK until the key clicks into the ON (down) position to produce all upper-case letters.

With CAPS LOCK on, the keys for numbers and symbols are not affected. To produce numbers and symbols printed on the bottom portion of the keys, depress the required key.

Examples:

With CAPS LOCK in ON position:

1 Pressing A will produce uppercase A.

2 Pressing 8 will produce number 8.

To produce symbols printed on the top portion of keys, depress either SHIFT key in conjunction with the required key:

3 SHIFT – 2 will produce the @ symbol.

4 SHIFT – ; will produce the : symbol.

To unlock the CAPS LOCK, simply depress that key again until it clicks into the OFF (up) position.

Figure 0

Note the difference between upper-case letter O and figure zero, ie 0. If you type a number containing a zero, you must use the figure 0.

Figure 1

The figure 1 must be used for numerical work. Some typists use lower-case L to indicate a 1 — the computer will not accept this.

Special keys and their uses

RESET	Do not use RESET unless instructed to do so. Will always stop computer operation. May cause a language prompt to appear.
RETURN	Normally used to indicate to the computer that you have finished typing an instruction. Also moves the cursor to the beginning of the next line.
CONTROL	Used in conjunction with another key to perform additional functions. Does not produce visible display. (More about this later.)
ESCAPE	Used before another key to perform additional functions. (More about this later.)
REPEAT	Every key has a repeat function. Depress and hold any key to see this function.
LEFT ARROW ←	Moves the cursor back one or more spaces and wipes out the characters over which it has passed. Although those characters still remain on the screen, they have been erased from the computer memory. Use this backspace key to position the cursor over the incorrect character, then simply retype the correct character.
RIGHT ARROW →	Moves the cursor forward one or more spaces. If the cursor passes over a character, that character will be re-entered without your touching the relevant key, and it will be replaced in computer memory.
DOWN ARROW ↓	Moves the cursor down one line, without affecting typing.
UP ARROW ↑	Can be used to move the cursor up one line without affecting typing. Not available in Applesoft BASIC.
TAB →|	Moves the cursor to the next tab setting. Tabs are preset every 8 spaces, or may be set by the operator. Not available in Applesoft BASIC.
DELETE	Not available in Applesoft BASIC.
SPACE BAR	Enters one blank space and moves cursor one space to the right. When SPACE BAR is used to space over previous typing, it removes those characters from the screen and from the computer memory.
OPEN APPLE 	Used in conjunction with other keys to restart the system when the power is on.

SOLID APPLE 

Used in conjunction with other keys to start an in-built self-test, which checks that the computer system is operating correctly.

Depress and hold SOLID APPLE.
Press CTRL–RESET. Release CTRL–RESET.
Release SOLID APPLE.

If the system is operating correctly, the message KERNEL OK will appear in 20–30 seconds.

If a different message appears, the computer requires service.

Press CTRL–RESET twice to return to BASIC.

Keyboard practice

Type the following paragraphs (1–5) to practise keyboarding until you become familiar with the different touch required. Read the paragraphs before you begin typing.

1 For your keyboarding practice you do not need to use RETURN. As you type, the cursor moves across the screen and then continues on to the next line, ie you do not need to press RETURN to go to the next line.

   If you accidentally press RETURN during your keyboard practice, the message SYNTAX ERROR will appear on the screen. Ignore this message during your keyboard practice.

2 The first line of typing commences at the bottom of the screen and as further characters are entered the display gradually moves up the screen. This is called **vertical scrolling**.

3 If you make a typing error, remember to use the LEFT ARROW key to backspace to the incorrect letter, then type in the correction, then use the RIGHT ARROW key to automatically retype characters previously cancelled.

4 The screen will display 40 characters across the screen and 24 lines down the screen. (If your computer has an 80 column card installed, the screen will display 80 characters across the screen.)

5 If you type more than about 240 characters, the computer will start to *beep*, then will place a backslash (\) on the cursor position and move the cursor to the next line to allow you to start typing again.

When the screen is full, try the next section.

How to clear the screen

Method 1

- Depress ESC key then release, type @ (SHIFT – 2) .
- Screen is totally cleared.

- Cursor has returned to top left of the screen which is called **home** position.

Type the previous section again to continue keyboard practice until the screen is filled. Then try Method 2 to clear the screen.

Method 2

- Depress RETURN to position cursor at left margin.
- Type HOME press RETURN.
- Screen is totally cleared.
- Cursor has returned to top left of screen (home position).

Note: RAM is not affected when screen is cleared.

Some editing features: how to correct errors

Type the previous section to improve your keyboard skills and practise these editing features.

LEFT ARROW	Moves cursor to left to allow correction of typing errors. Characters passed over by the cursor are erased from memory, but not from screen.
RIGHT ARROW	Moves cursor to right. Automatically retypes the characters passed over by the cursor, and replaces them in memory.

Use LEFT ARROW to position cursor over error. Type correction. Use RIGHT ARROW to replace characters in memory and return to original typing position.

CONTROL–X	Press and hold CTRL, type X . Places a backslash (\) at cursor. Cancels the whole line from memory, but not from screen.

Cursor movement

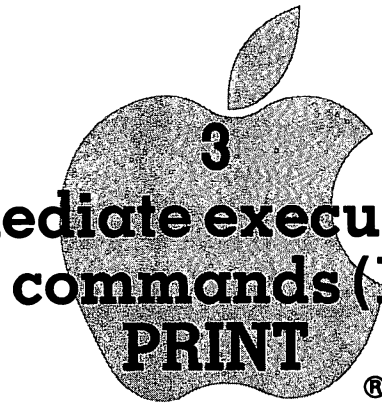
When the cursor reaches:

- the top of the screen — it will stop
- the bottom of the screen — the cursor remains stationary, but the printing on the screen moves up
- the right edge of the screen — with further moves right it will reappear at the left margin one line lower
- the left margin of the screen — with further moves left it will reappear on the right edge, one line higher.

Clearing portion or all of screen display

The commands below will remove a portion or all of the screen display. Try each command in conjunction with some of the **pure cursor** moves you have learned.

ESC–E	Depress ESC and E together. Clears all characters from cursor to end of line, from screen and memory.
ESC–F	Depress ESC and F together. Clears all characters from cursor to end of screen, from screen and memory.
ESC @	Do you remember this one? Depress ESC, release, depress @ (SHIFT–2). Clears entire screen. Cursor returns to home position.
HOME	You should remember this one too! Depress RETURN, type HOME , depress RETURN. Clears entire screen. Cursor returns to home position.



3 Immediate execution of commands (1) PRINT

The functions explained in this chapter are available under the directions of the programs permanently stored in ROM. All data and/or instructions entered at the keyboard will be temporarily stored in RAM. You will not need to use either a disk or a cassette.

When you were practising your keyboarding skills, you did not have to use the RETURN key at the end of each line. If you accidentally pressed RETURN, the message SYNTAX ERROR appeared on the screen, which you were told to ignore.

In fact, the RETURN key has a special function — it is a signal to the computer that you have finished typing an instruction. In Immediate Execution the computer must now obey your instructions.

A word of warning though — if you type an instruction, press RETURN, and the message SYNTAX ERROR appears, the computer is telling you: 'You have made an error — I cannot read your instruction'. If this happens, check your instruction carefully and then retype correctly.

If you notice a typing error before you press RETURN, use the editing keys to correct the error.

Call up prompt and cursor.

PRINT command

Note: Commands must be typed in *all UPPER-CASE letters.*

1 Type PRINT "HELLO" then press RETURN.

2 The computer has followed your instructions — it has printed the word HELLO.

Note: Any characters enclosed in quotation marks will be printed exactly by the PRINT command. The first quotation mark indicates that the following character is the start of the material to be printed, while the

second quotation mark indicates that the material to be printed has been completed.

3 Type PRINT "My name is _____." Press RETURN.

4 Try PRINTing some more words and short sentences, remembering to press RETURN when you have finished typing your instruction.

You should understand the function of the RETURN key by now, so the instruction — press RETURN — will not be shown in the following instructions.

5 What happens if you forget to type the first quotation mark? Try it.

Type PRINT YOUR NAME"

The result — 0 — the computer could not read your command as there was no opening instruction.

6 Type PIRNT "YOUR NAME" — deliberately misspelling PRINT to see the computer's reaction.

The result — SYNTAX ERROR — computer could not read your instruction.

7 Type PRINT "1234" Result?

8 Type PRINT 1234 Result?

Note: Numbers (numeric data) will be printed exactly whether enclosed or not enclosed in quotation marks.

9 Type PRINT "30 JUNE 1984" Result?

10 Type PRINT "30/6/1984" Result?

11 Type PRINT 30/6/1984" What happened?

The computer reads a / as meaning 'divide by', so in this example 30 was divided by 6, giving 5, and 5 was divided by 1984.

Note: To print an exact copy of a mixture of numerics and non-numerics (string data), all characters must be enclosed in quotation marks.

Practise this section until you feel confident about PRINTing an exact copy of any characters.

Abbreviated PRINT

When you are using Applesoft (prompt]), the PRINT command can be shortened to a question mark.

PRINT "January" (and RETURN of course)
? "January" Was the result the same?

From now on, if you are using Applesoft, you may use either the word PRINT or the question mark.

Multiple PRINT

Applesoft allows you to use the PRINT command more than once in the same line. Try this example:

PRINT "Monday" : PRINT "Tuesday" Result?

Note: The colon is used to indicate a following PRINT.

Length of PRINT statement

You can instruct the computer to PRINT a longer message. However there is a limit to the length of the message. If it contains more than about 240 characters the computer will interrupt your typing with a *beep* and a backslash, and move the cursor to the next line.

In Applesoft, the computer will PRINT up to about 240 characters if you stop typing before the backslash appears, *but,* if the backslash has appeared your statement will not be printed at all. In fact, you will get a SYNTAX ERROR message to let you know that the computer is unable to follow your instruction.

Try this yourself, to see the computer's reaction. Use the first paragraph of this section, which contains more than 240 characters, as your message. Remember that you must not use the RETURN key to continue typing on the next line.

PRINT "You can instruct the computer . . ."

Methods of display

There may be occasions when you want to space your work across the screen, or close-print your work. Try the following methods.

1 PRINT "HEADING", "HEADING", "HEADING"
The comma instructed the computer to print the words in columns.

2 PRINT "Monday"; "Tuesday"; "Wednesday"
The semicolon instructed the computer to print the word in the next space.

How could you get a space between the words so that they do not run together?

Try PRINT "Monday "; "Tuesday "; "Wednesday "

By leaving a space after the last letter of each word, the printing will also leave a space.

3 PRINT TAB (5); "Monday"; TAB (20); "Tuesday"
The word TAB instructed the computer to PRINT the words at the character position on the line indicated by the figure enclosed in brackets. Notice that the semicolon must be used after the TAB input.

Practise these methods with some examples of your own.

Combining some PRINT instructions

So far, you have found out that:

- PRINT or ? are commands to the computer to print your message.
- The comma is an instruction to print in columns.
- The semicolon is an instruction to close-print.
- The colon allows more than one PRINT command in a line.
- The TAB command instructs printing at a specified line position.

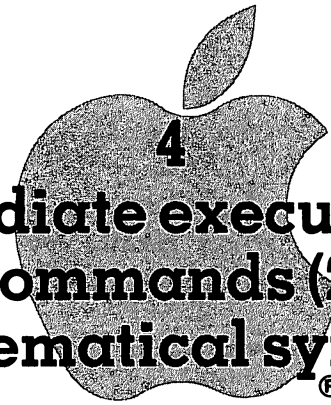
Let's try combining some of these instructions.

```
1 PRINT "EXAMPLES ", "OF ", "DISPLAY"
2 PRINT TAB(5); "EXAMPLES", "OF", "DISPLAY"
3 PRINT TAB(5); "EXAMPLES"; TAB(20); "OF"; TAB(30);
  "DISPLAY"
4 PRINT "EXAMPLES" : ? "OF" :? "DISPLAY"
```

Try some of your own combinations and also some longer messages. Think carefully about the display you want, and make sure that you give the computer the correct instructions.

Summary

- Any characters enclosed in quotation marks will be printed exactly.
- Numbers (numeric data) will be printed exactly whether enclosed or not enclosed in quotation marks.
- A mixture of numerics and non-numerics (string data) must be enclosed in quotation marks.
- The PRINT command can be abbreviated to a question mark.
- The PRINT command may be used more than once in the same line by using a colon.
- The length of a PRINT statement must not be more than about 240 characters.
- The comma is an instruction to print in columns.
- The semicolon is an instruction to close-print.
- The TAB command instructs printing at a specified line position.



4 Immediate execution of commands (2) Mathematical symbols

By the time you start this chapter, you already know that you must type PRINT or ? to command the computer to print your message. So from now on the command PRINT will not be shown.

Remember that you are still working under the direction of permanent programs in read–only memory. Keyboard entries are temporarily stored in random–access memory.

Call up prompt and cursor.

Symbols for mathematical functions

The computer can be used to supply answers to mathematical problems. Use the following symbols:

+	addition
−	subtraction
*	multiplication
/	division
Λ	exponentiation (eg squaring)
Λ (1/2)	square root of . . . (or exponentiation to the 1/2 power)

When you are typing the following problems, you can:

- leave a space between characters, eg 4 + 4
- leave no space between characters, eg 4+4

Either way, the computer will read your instructions.

Addition

1 4 + 4 (Did you remember PRINT and RETURN?)

2 75 + 17

Note: The computer will supply only the answer. To make the result more meaningful, type:

"75 + 17 = "; 75+17 Result 75 + 17 = 92

If you use a comma after the closing quotation mark, the answer will be printed in the next column,

eg"75 + 17 = ", 75 + 17

3 "110 + 78 + 97 = ", 110+78+97

What happens if you leave out the first quotation mark?
Try it and see.

The message you received (TYPE MISMATCH ERROR) tells you that the computer cannot understand your instruction.

4 "4 + 6 = ";4+6

What happens if you enter the wrong figures?
Try this:

"4 + 5 = ";4+6

Result: 4 + 5 = 10 which you know is wrong!

But the computer followed your instructions by:

a PRINTing the quoted figures 4 + 5 =
b Adding the numbers 4+6 10

So you must be very careful in entering data — always check your typing before pressing RETURN.

Try the following:

Subtraction

1 117 − 36
2 "42 − 19 = ";42−19
3 "176 − 28 − 43 = ";176−28−43

Multiplication

1 42 * 6
2 "17 * 31 = ";17*31
3 "82 * 4 * 12 = ";82*4*12

Division

1 1776/12
2 "273/13 = ";273/13
3 "3976/4/7 = ";3976/4/7

Squaring or cubing a number

1 5∧2 ie 5* 5
2 "6∧3 = "; 6∧3 ie 6 * 6 * 6

Finding the square root or cube root

1 25 Λ (1/2) ie square root of 25

2 It is easier to use the SQR function, eg SQR (49) SQR (23104)

3 216 Λ (1/3) ie cube root of 216

Using decimals

Applesoft allows decimal numbers to be used in all calculations.
There are some points you should remember though:

- If you add 10.14 and 2.16 the answer is 12.30.
  The computer will print 12.3 ie it will not print trailing zeros.
- If you add 033.62 and 021.20 the answer is 054.82.
  The computer will print 54.82 ie it will not print leading zeros.

Rounding in Applesoft

If you type a number with more than nine digits, or if the answer to a problem contains more than nine digits, the computer will round the number to nine digits.

Example

Type PRINT 2345.67891234

Result 2345.67891 Rounded down because tenth digit was less than 5.

Type PRINT 2345.67898933

Result 2345.67899 Rounded up because tenth digit was equal to or greater than 5.

Scientific notation

If you type in a long number, the result may be shown with an included letter E, eg Type PRINT 1234567899876543 Result?

This is called scientific notation.
Follow these examples:

1 Type PRINT 0.00356 will display as 3.56E−03

To get the correct answer, look at the symbols following the character E.

The minus symbol (−) means 'move the decimal point to the left'; the numbers (03) mean '03 spaces'. So the correct answer would be 0.00356.

2 Type PRINT −9876543210 will display as −9.87654321E+09

The plus symbol (+) means 'move the decimal point to the right'; the numbers (09) mean '09 spaces'. So the correct answer would be −9876543210.

Unless you understand scientific notation, use simpler calculations until you feel confident with the system.

Mathematical functions

There are a number of mathematical functions available and if you intend to use the computer for more complex mathematical problems you should check the reference manuals for a complete listing.

Try these three examples:

Square root SQR(169)

Integer (whole number)

Returns largest integer less than number INT(16.7)
INT(16.2)

Note: By adding .5 to the number in brackets, the **integer** function can be used to round (up or down) to the nearest whole number.

INT(16.7+.5)
INT(16.2+.5)

Random number

Returns a different number each time it is used. Work each example at least twice. RND(5)
RND(20)

Relational operators

Relational operators can be used to compare data and to determine whether the data is the same value, or a different value.

Use the following symbols:

<	less than
>	greater than
=	equal to
<> or ><	not equal to
<= or =<	less than or equal to
>= or =>	greater than or equal to

Try these examples. Remember to use PRINT and RETURN.

1 10 = 10 Result 1 which means TRUE
2 10 = 11 Result 0 which means FALSE
3 10 > 8
4 98 <> 98
5 78 =< 97
6 1983 => 1982
7 "MONDAY" = "MON"
8 "THEIR" = "THERE"

Logical operators

Logical operators test the comparison between two or more relations. The logical operators are:

AND OR NOT

Use PRINT.

1 AND results in a 1 (true) only if both values are true
eg 1 AND 1 = 1 results in 1 (true)

2 OR results in a 1 (true) if either value is true
eg 2 OR 3 = 2 results in 1 (true)

3 NOT complements each value
eg NOT 7=6 results in 0 (false)

Combinations of symbols

Addition and subtraction

202 − 57 + 143 − 98

Calculations are performed from left to right.

Addition, subtraction and multiplication

202 + 57 − 25 * 2

The multiplication is performed first. Then additions and subtractions are performed from left to right.

Addition, subtraction, multiplication and division

987 + 17 − 30 * 2 + 88/11

The multiplications and divisions are performed first, from left to right. Then additions and subtractions are performed from left to right.

Addition, subtraction, multiplication, division and exponentiation

10 Λ 3 + 1079 + 10 * 6 − 31/16 − 4 Λ 2

Exponentiation (eg squaring) is performed first, from left to right. Then multiplications and divisions are performed from left to right. Then additions and subtractions are performed from left to right.

The computer has definite rules for performing calculations. These rules are called **order of precedence.**

Order of precedence

Order of precedence determines the order of the steps for all mathematical calculations.

Step 1 () Any calculation enclosed in brackets, from left to right.

Step 2	−	Negative numbers.
Step 3	Λ	Exponentiation from left to right.
Step 4	*/	Multiplications and divisions from left to right.
Step 5	+ −	Additions and subtractions from left to right.

Rule: If there is more than one calculation using the same symbol, calculations are performed from left to right.

Study this example and follow the steps:

	− 2 + 20 + 120 + 6/3 − 4 * (2 + 10) + 6 Λ 2
Step 1 Brackets	− 2 + 20 + 120 + 6/3 − 4 * (2 + 10) + 6 Λ 2
Step 2 Negative no	− 2 + 20 + 120 + 6/3 − 4 * 12 + 6 Λ 2
Step 3 Exponentiation	18 + 120 + 6/3 − 4 * 12 + 6 Λ 2
Step 4 Multiplication and division	18 + 120 + 6/3 − 4 * 12 + 36
Step 5 Addition and subtraction	18 + 120 + 2 − 48 + 36

Getting the right answer every time

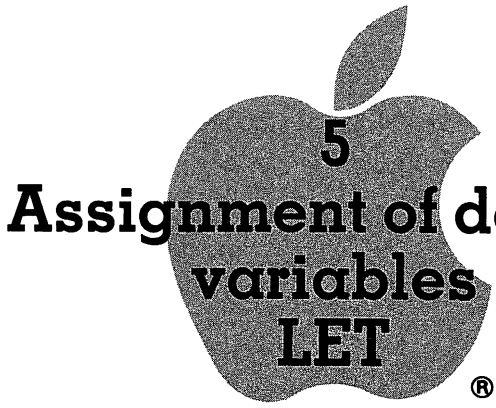
Work these examples yourself (without the computer!) and then check your answers on the computer.

6 * 6/2 + 4 * 6 Λ 2 + 10
10 * 4 * − 2 + 10 * 5 Λ 3

Notice the different results in the next two problems because of the inclusion of brackets.

10 + 2 * 4 − 6/3 = (16)
(10 + 2) * (4 − 6/3) = (24)

Now work some examples of your own—without the computer—and then check your answers on the computer.



Call up prompt and cursor.
Remember: ROM contains the programs to process your instructions. RAM contains the data and/or instructions entered via the keyboard.

Pigeonholes or storage locations or addresses in memory

Every computer has the capability of storing information for later use. Using the Applesoft language, there are hundreds of pigeonholes into which data can be placed and recalled later. You, as the operator, can name each pigeonhole so that you will know where to find any data.

In this section you will use the letters A B C D E F as names for six pigeonholes.

Storing data in memory

Storing data in memory means inserting a specific data item into the named storage location in memory.

Do not use PRINT. The word LET must be typed in UPPER CASE.

Type LET A = 344 (and of course RETURN)
LET B = 125
LET C = 22

The word LET is optional. Type the following data without using LET.

D = 5
E = 55
F = 4

Each number has now been placed in its own pigeonhole.

Retrieving data from memory

Instruct the computer to PRINT the contents of each pigeonhole.

PRINT A,B,C
PRINT D,E,F All correct?

Changing data in memory

Type A = 225 PRINT A

What happened? The previous data (344) has been removed from the pigeonhole and cancelled from memory, and has been replaced by the new data (225).

Performing calculations with data in memory

Instruct the computer to carry out these tasks. Remember the command PRINT and RETURN. Use the mathematical symbols you have already learned.

1 A plus B plus C
2 E minus D minus F
3 A multiplied by E multiplied by C
4 B divided by D
5 A plus D multiplied by F
6 A plus B divided by D
7 D squared
8 Square root of B
9 C multiplied by (E plus A)
10 A minus (F multiplied by D) plus (B plus C)

After doing the calculations, check the contents of the pigeonholes again—they should be the same as before.

PRINT A,B,C
PRINT D,E,F

Try some calculations of your own, using the data already stored in the pigeonholes.

Clearing data from memory

When you have finished your calculations, you can clear all the pigeonholes and set them to zero by typing CLEAR or NEW .

Numeric variables

The correct name for pigeonholes is **variables**. This is the term that will be used from now on.

In the previous examples you used a letter to identify each variable. All the data you used was numeric, ie numbers only. When the data is all numeric, you must use a **numeric variable.**

Numeric variables:

- must begin with a letter (A–Z)
- may be followed by:
  another letter (A–Z)
  or a digit (1–9)
  eg A = 10
  LZ = 20
  J2 = 30

String variables

What happens if you want to store data that is not all numeric?

Data that is not all numeric is called **string data**. It consists of a string of letters (alphabetic) or a mixture of letters, numbers and symbols (alphanumeric).

String data must use a **string variable**, and must be enclosed in quotation marks.

String data may be entered in either upper or lower case letters.

String variables

- must begin with a letter (A–Z)
- may be followed by:
  another letter (A–Z)
  or a digit (1–9)
- must end with a dollar sign ($)
  eg A$ = "NAME"
  CM$ = "Place"
  W4$ = "data"

Using numeric and string variables

Try these examples.

1 Type A$ = "STRING DATA " (and RETURN of course)
B$ = "IN QUOTATION MARKS"
PRINT A$; B$

Did you notice that there is a space after DATA and before the quotation marks?

2 Type
```
A = 11
B = 14
M$ = "MARY IS "
R$ = "ROGER IS "
Y$ = " YEARS OLD"
PRINT M$; B; Y$
PRINT R$; A; Y$
```

Try some examples of your own, combining numeric and string variables. When you have finished, remember NEW or CLEAR to clear all variables.

6 Deferred execution of commands (1) Program execution

Deferred execution simply means that you are going to input a number of instructions to be performed later. This is really writing a simple computer program.

The major difference between a program written for deferred execution and statements written for immediate execution is that in deferred execution each statement must be preceded by a number.

Call up prompt and cursor.

Sample program

1 Input as shown, inserting your own personal details in statements 30, 40, 50 and 60.

```
NEW

10  REM  MY FIRST PROGRAM
20  PRINT "EXERCISE NO. 1"
30  PRINT "YOUR NAME"
40  PRINT "NUMBER AND STREET"
50  PRINT "SUBURB
60  PRINT "PHONE NUMBER"
9999  END
```

LIST (watch computer reaction to this instruction)

RUN (watch computer reaction to this instruction)

2 What happened?

The computer followed your instructions by:

a storing your statements (numbered 10 to 9999) in RAM

b LISTing your statements for you to check

c RUNning your instructions under the direction of programs stored in ROM.

3 Let's look closely at each part of your program.

NEW

NEW is an instruction to the computer to wipe out previous data stored in memory (ie clear all variables) and to store a brand new program.

Statement numbers, eg 10 to 9999

The computer stores statements in ascending order, ie lowest to highest. It is always a good idea to increase statement numbers by 10 (as you did), in case you have to insert extra statements when you are checking the program.

REM

REM is short for REMARK. REM statements are for reference purposes only—the computer does not act on them. They are helpful to identify and clarify your program.

END

END signifies to the computer the END of the program. It must always have the highest statement number.

In Applesoft, the highest statement number you can use is 63999. For your exercises, use 9999 to signify END.

Note: With the Apple//e it is not essential to use an END statement. However, it is good programming practice to do so.

LIST

LIST is an instruction to the computer to LIST your program on the screen so that you can check it.

When you typed LIST the whole program was listed on the screen. It is also possible to LIST portions of the program:

LIST	Displays whole program.
LIST 40	Statement 40 is displayed.
LIST 30, 50	Statements from 30 to 50 are displayed
LIST ,40	Statements from beginning to statement 40 are displayed.
LIST 40,	Statements from statement 40 to END are displayed.

RUN

RUN is an instruction to the computer to obey your program instructions.

When you typed RUN the whole program was executed on the screen. You can also RUN portion of the program:

RUN	Entire program executed.
RUN 30	Program executes from statement 30.

Inserting a statement

Your program (exercise no 1) is still stored in computer memory.

Type LIST to display the program on the screen.

Now you are going to insert three additional statements in your program. Two statements will be to insert a blank line after "Exercise No. 1" and after "Your Name". The third insert will be your own postcode and this will be positioned after your suburb.

Type (at the cursor position):

```
25  PRINT
35  PRINT
55  PRINT "POSTCODE"
```

LIST

Notice how the computer has reorganised the lines in ascending statement number.

RUN

Changing a statement

If you want to change the contents of a statement (or correct an error) you may use either of the following methods.

To renumber your program as exercise no 2:

Method 1: retype the entire statement.

Type:

```
10  REM  MY SECOND PROGRAM
20  PRINT "EXERCISE NO. 2"
```

Method 2: **pure cursor moves.**

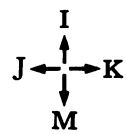
If you do not wish to retype the statement, you may use pure cursor moves.

This group of pure cursor moves is easy and fast if you want to move the cursor more than once in any direction to make corrections. These moves do not affect display or memory.

Pressing ESCAPE places the computer in EDIT mode.

In EDIT mode the direction keys (K, J, M, I) can be depressed once, or depressed and held, for rapid movement.

To finish EDIT mode, press SPACE BAR, then make correction.



1 ESC K—to move cursor to *right*.

- Depress ESC, release (now in EDIT mode).
- Depress K (hold to engage repeat function).

- Cursor will move to right.
- When cursor at required position, press SPACE BAR to finish EDIT mode.

2 ESC J—to move cursor to *left*.
- Depress ESC, release (now in EDIT mode).
- Depress J (hold to engage repeat function).
- Cursor will move to left.
- When cursor at required position, press SPACE BAR to finish EDIT mode.

3 ESC M—to move cursor *down* the screen.
- Depress ESC, release (now in EDIT mode).
- Depress M (hold to engage repeat function).
- Cursor will move down the screen.
- When cursor at required position, press SPACE BAR to finish EDIT mode.

4 ESC I—to move cursor *up* the screen.
- Depress ESC, release (now in EDIT mode).
- Depress I (hold to engage repeat function).
- Cursor will move up the screen.
- When cursor at required position, press SPACE BAR to finish EDIT mode.

To make corrections

1 LIST statement in which correction is to be made.
2 Depress ESC (to enter EDIT mode).
3 Depress I until cursor is positioned on listed line.
4 Depress J until cursor is positioned on first digit of statement number.
5 Depress SPACE BAR to finish EDIT mode.
6 Use RIGHT ARROW key until cursor is positioned at correction point.
7 Type correction.
8 Use RIGHT ARROW key to move to end of statement.
9 Depress RETURN.

LIST to check correction.
RUN.

Note: To cancel character/s within quotation marks, use pure cursor move ESC K instead of RIGHT ARROW key. The character/s will not be replaced in memory.

Deleting a statement

If you want to DELETE portion of your program, eg to remove the exercise number:

LIST or LIST 20

Then type:

20 (this will delete the previous statement 20)

Next: LIST (statement 20 has been deleted)

RUN

If you want to delete several successive lines, eg your address:

LIST or LIST 40,55

Then type:

DEL 40,50 (statements 40 to 55 inclusive will be deleted)

Next:

LIST (statements have been deleted)

RUN

Clearing the screen

You have already learned how to clear the screen in immediate execution of commands. You can also clear the screen in deferred execution.

Type LIST (what's left of your program is displayed!)

Now type

70 HOME

LIST (to check)

RUN

Screen is totally cleared.

Is your program still in memory? Type LIST to find out.

You can also clear the screen by typing:

HOME or ESC @ (SHIFT–2)

Clearing the memory

The only way to clear your program from memory is to type:

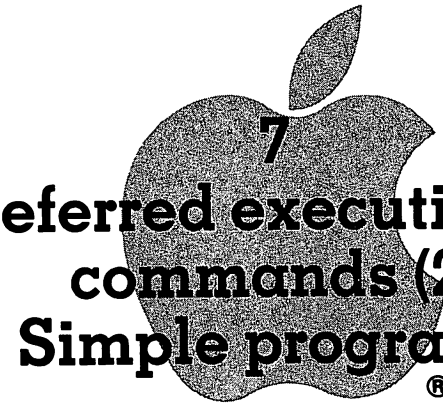
NEW

Now try

LIST (program has been wiped from memory)

Readability

Did you notice that when a program is LISTed, the computer has inserted additional spaces to make it easier to read?



7
Deferred execution of commands (2) Simple programs

Now you are going to input, LIST and RUN a NEW program, make alterations to that program and then LIST and RUN the amended program.

Exercise No 3

Enter this program:

```
NEW
10  REM  SIMPLE CALCULATIONS
20  PRINT "EXERCISE 3"
30  PRINT "**********": PRINT
40  PRINT "YOUR NAME": PRINT
50  A = 50:B = 10:C = 12
60  PRINT "A = ";A: PRINT
70  PRINT "B = ";B: PRINT
80  PRINT "C = ";C: PRINT
90  PRINT "A + B * C = ";A + B * C
100  PRINT
110  PRINT "A/B * C = ";A / B * C
9999  END
```

LIST and RUN

If you leave out an instruction, eg PRINT, and then try to RUN the program, you will receive a SYNTAX ERROR message, identifying the statement by number. Check that statement number in your LISTed program and make the necessary correction. Try this yourself, by deliberately leaving out the word PRINT in statement 60.

Now make these alterations to your program:

Change A to 100, B to 20 and C to 24.
Delete statements 60 to 80.

LIST and RUN

Exercise no 4

This program works out the cost of buying 48 articles at $17.58 each. See if you can understand the steps, then enter the program.

NEW

```
10  REM  TO DETERMINE COST
20  PRINT "EXERCISE NO. 4"
30  PRINT "**************": PRINT
40  PRINT "YOUR NAME": PRINT
50  REM   N = NUMBER OF ARTICLES
60  REM  P = PRICE OF EACH ARTICLE
70  REM   C = TOTAL COST
80  PRINT "NUMBER","PRICE","COST $"
90  N = 48:P = 17.58:C = N * P
100  PRINT N,P,C
9999  END
```

LIST and RUN

Exercise no 5

Make the following changes to exercise no 4:

Change to exercise no 5
Delete statements 40 and 50
Insert a blank line after statement 90
Change N to 75
Change P to 25.99

LIST and RUN

Exercise no 6

Design a simple program to solve this problem:

If you purchased 32 articles and the total cost was $497.60, what was the price of each article?

Hint: You should be able to adapt exercise no 5 to solve this problem.

LIST and RUN

Exercise no 7

Design a simple program to find the area of a rectangle where L = length, W = width and A = area.

Your program should include statements, such as the following, to print:

AREA OF RECTANGLE IS — sq.cm.

Hint: You may be able to modify your previous program.

LIST and RUN

Inverse, flash and normal

Unless you have turned the computer off, your exercise no 7 is still stored in memory. Let's try something different!

1 Clear the screen. Remember: HOME.

2 Type INVERSE .

3 Type LIST .

Notice the difference in the visual appearance of the program?

4 Type HOME .

5 Type FLASH .

6 Type LIST .

What difference does this make to the visual appearance?

7 Type NORMAL to return to normal output.

8 Type LIST .

8 Disk drive and disks

Omit this chapter if you are using a cassette recorder and cassettes.

Using the disk drive and disks is a convenient and efficient method of accessing a large amount of data and of providing secure storage for your own programs.

Disk drive

The operation of the disk drive can be compared to that of a cassette recorder. Look at the following steps:

Cassette recorder	*Disk drive*
Insert cassette	Insert disk
Close door	Close door
Depress PLAY button	
Head lowered on cassette	Head lowered towards disk (but not actually touching)
Sound is produced	Contents are 'read'

Disks

Disks are used for storage of data and retrieval of this data for future reference. The data is filled under **FILENAME**, ie surname, subject, etc.

Make-up

A disk is a small 12.5 cm plastic disk, coated so that information may be magnetically stored on it and if desired erased from its surface — the coating is similar to that on a recording tape/cassette. The disk is sealed in a square plastic cover to protect it. The cover helps to keep it clean and

allows it to spin freely in the disk drive. *The cover is never opened.*

The disk you will be using will probably be a 'floppy' disk. This means that it is flexible — but bending will damage it. The protective cover contains both lubricants and cleaning agents. Always treat a disk with respect.

Some disks have a reinforced section around the centre spindle hole. This helps to prevent excessive wear on the inner edge of the disk.

There is a slot in the cover through which the 'head' is able to read the contents.

A disk can store almost a million bits of information — each individual bit occupies a very small portion only — so an invisible scratch or even a fingerprint can cause an error.

Care

Never let anything (especially fingers) touch the brown/grey surface of a disk — handle by the plastic cover only.

Average life

The average life of a disk is 40 hours. Considering the few seconds it takes to read the information, a disk should last a long time.

Storage

Keep disks in paper jackets when they are not in use, to minimise static electricity build-up which attracts dust. It is preferable to store them vertically.

Keep them away from magnetic fields, eg electric motors, magnets, etc. *Do not place disks on top of the computer.*

Keep disks out of the sun and away from heat. They can warp, and the first evidence of heat damage is a warped or bent plastic cover.

Label

Always label disks so you will know the contents. Write the title on a label and then attach it to the disk. Do not write on a label already attached to a disk, as this could damage the disk.

Write-protecting

You will notice a small 'cut-out' on the side of a disk cover. This may be covered with a tab so that the recorded contents of the disk cannot be erased. This is very similar to a cassette where the two small lugs at the back are removed so that nothing can be recorded over and hence nothing can be erased. A disk that you intend to work on should have this tab removed, so that information can be recorded on and read from the disk.

Inserting a disk

1 The red IN USE light on the front of the disk drive must be OFF.

2 Hold the disk carefully in a horizontal position without touching the exposed sections. The label should be facing up. This means that the write-permit notch (or write-protect tab) will be to the left as the disk is inserted. You will feel the disk seat into place.

3 Close the drive door — this actually lowers the head towards the disk so that it can read the contents. However, the head does not actually touch the disk.

4 The red IN USE light on the drive will light up while the motor is working. The data and/or instructions on the disk will be transferred to RAM. **Do not attempt to use the machine while this red light is on,** ie do not open the disk drive door and do not use the keyboard.

Catalog

1 Insert System Master Disk DOS 3.3 in Drive 1. DOS stands for Disk Operating System.

This disk will put an additional language into the computer's memory and will also load the programs recorded on the disk. This is sometimes referred to as **booting the system**.

Each time the computer is switched off you will have to boot the system when restarting.

2 Turn machine ON, and immediately close disk drive door.

3 Red IN USE light on disk drive will light up while the head is reading.

4 Wait until the red IN USE light goes off.

5 To see what programs are on the disk, type CATALOG immediately after the cursor. Press RETURN. This is an instruction to the computer that you wish to see a listing of the programs — similar to using an index in a book.
Note: Americanised spelling of CATALOG.

6 A list of filenames will appear on the screen — this is often referred to as a **directory.**

7 Press SPACE BAR to see if there are further filenames in the directory which did not initially show on the screen.

8 The catalog contains details of the files stored on the disk. The first column identifies the language for a file.

A = Applesoft BASIC	I = Integer BASIC
B = binary	T = text

An asterisk (*) before the letter means that the file is **locked**, ie it cannot be changed or erased.

The second column shows the amount of space (in sectors) which the file occupies.

The third column shows the name of the file.

Using files on a disk

1 To list programs, type LOAD filename . Wait until the red IN USE light goes off. Then type LIST .

2 To run programs, type RUN followed by filename.

3 Call up Color Demosoft by typing RUN COLOR DEMOSOFT . Press RETURN. Be careful to copy exactly the spelling and spacing.

An **inner menu** will appear on the screen — this is a menu that relates to this particular program only.

Choose each number in turn — following the instructions on the screen — and the various graphics will be displayed. When making a selection always use the number.

4 Press RETURN to stop demonstration and to return to the inner menu.

5 Press CTRL–RESET and then type CATALOG to return to the catalog.

Phone list

This program will allow you to follow a set of instructions and will show you how data which you enter is stored and can be retrieved when desired — on the screen or on the printer.

1 Select Phone List. Type RUN PHONELIST (leaving no space between the words). What did you notice?

FILE NOT FOUND — because the filename was not typed *exactly* as shown on the directory the computer could not locate it.

2 Type RUN PHONE LIST .

3 Inner menu, as shown below, will appear on screen:

1 Search for a listing
2 List entire file
3 Enter new listings
4 Delete a listing
5 Edit a listing
6 Printer ON/OFF
7 Exit this program

Enter new listing

4 Select number 3.

5 Enter name: JACKSON HARRY .
Press RETURN.

6 Enter number: be sure to copy format displayed. 075 208 9432
Press RETURN.

7 Screen will display total entry and will ask if entry is correct. Y/N — if correct type Y and press RETURN.

8 Computer will return to inner menu.

9 ENTER these five names to the phone list:

DR KING KEVIN	032 124 8735
SMITH BRIAN	071 341 6825
JACKSON SHERYL	062 731 8240
JACKSON COLIN	049 356 7842
DR BATT ROBERT	052 246 8024

List entire file

10 Select number 2.

11 All names and phone numbers will appear on the screen.

12 Return to inner menu.

Search for listing

13 Select number 1.

14 When searching for a file the computer can search by word or by character.

15 Select 'word'.

16 ENTER SEARCH KEY will appear on the screen. The computer will search for all entries beginning with this particular letter.

17 Type J — it will locate all entries where the first letter is J.

18 Repeat step 13.

19 Select 'character'.

20 Enter SEARCH KEY — DR — the computer will now search (character by character) for all entries where the first two characters are DR.

(Listing for DR BATT and DR KING will appear on the screen.)

Delete a listing

21 Select number 4.

22 Type entry SMITH BRIAN .

23 Entry will be shown on screen. You will be asked to decide whether you wish to delete. Select yes (Y).

24 LIST the entire file to check entry has been deleted.

Edit — to make a change to an entry

Change the phone number for JACKSON COLIN to 049 243 4917.

25 As name is not changed, simply retype.

26 Insert new phone number.

27 Return to inner menu.

28 List the entire file to see changes.

Printer on/off

29 Select number 6. Select printer ON.

30 Prepare a printed list of the entire file.

Before proceeding check that the printer has a supply of paper inserted. See the manual for your particular printer for details of paper insertion.

31 Turn printer ON. When printing is completed turn printer OFF.

Exit this program

If nothing is recorded on disk press CTRL–RESET.

32 Return to directory — type CATALOG .

Choose any of the other programs

If you wish to see any of the other programs listed in the directory simply type RUN followed by filename.

Remember that if you turn the computer off you must boot the system before inserting this disk.

If using the Supermath program, when typing answers to addition, subtraction and multiplication the computer records units first, then tens, hundreds, etc.

Initialising a disk

A totally blank disk is of no use to your Apple//e. It needs to be set out magnetically in much the same way as you might draw lines on a blank page before you start to write.

So, the term **initialising a disk** simply means preparing:

- a totally new disk for use, or
- deleting all the contents of a previously used disk, so that it may be re-used.

Method

1 Load System Master Disk 3.3, and then remove disk.

2 Insert blank disk in Drive 1, close door, press RETURN.

3 CAPS LOCK ON.

4 Type NEW and then press RETURN.

5 Type 10 PRINT "HELLO" and then press RETURN.

6 Type INIT HELLO and press RETURN.

Initialising will start immediately and once the process is complete and the red IN USE light goes off, the disk is ready for use.

Please initialise a blank disk and then check the catalog.

Disk or file protection

You already know that you can only store data on a disk which has the write-permit notch uncovered.

To protect the entire contents of a disk, place a write-protect tab over the notch. This prevents information being written on to the disk.

To protect a file on a disk from alteration or erasure, use the LOCK command, eg Type LOCK filename .

LOCKed files are identified on the catalog by an asterisk in the first column.

You should now LOCK the "HELLO" file.

To UNLOCK a file, type UNLOCK filename .

Saving information on a disk

At this point you will learn how to save one of your simple programs on disk for future reference. If the machine has been switched off, remember to boot the system.

1 Insert the disk which you have just initialised.

2 Type NEW .

3 Enter a simple program, one of your previous exercises.

4 LIST and RUN the program.

5 Select a filename. The filename must:

- begin with a letter
- contain no more than thirty (30) characters
- contain no commas.

6 Type SAVE filename , eg SAVE EXERCISE 1 .
The data and/or instructions in your exercise will be copied from RAM to the disk.

Once you have SAVED the program, type CATALOG and press RETURN to see the name of the file listed on the disk.

You would then be able to recall exercise 1 from the disk at any later time by:

a LOAD EXERCISE 1 and then LIST

or **b** RUN EXERCISE 1

Stopping a program

Programs can be stopped by any of the following methods:

1 Press CTRL–C.

2 Press CTRL–RESET.

3 Press OPEN APPLE and CTRL–RESET (all data stored in RAM is lost).

Another method of booting the system

If you have been using the computer without engaging the disk drive, it is possible to boot the system without turning the computer off. Any data you had in memory will be lost.

1 Insert initialised disk in Drive 1.

2 Press and hold OPEN APPLE key. Then press CTRL–RESET.

3 Wait until the red IN USE light goes off.

Deleting a file from disk

Call up CATALOG.

Type DELETE followed by filename — press RETURN — program is automatically deleted.

Making a copy of an entire disk

Obtain a blank disk and a disk containing some of your own files. When making a copy you do not need to initialise the blank disk as the COPY program does this automatically.

With the System Master Disk DOS 3.3 in Drive 1, type RUN COPYA and press RETURN.

The screen will display:

ORIGINAL SLOT	DEFAULT = 6	press RETURN
ORIGINAL DRIVE	DEFAULT = 1	press RETURN
DUPLICATE SLOT	DEFAULT = 6	press RETURN
DUPLICATE DRIVE	DEFAULT = 2	press RETURN

Remove System Master Disk. Insert original disk in Drive 1 and blank disk in Drive 2. Press RETURN.

System will copy disk and then ask:

DO YOU WISH TO MAKE ANOTHER COPY? Press Y or N

Note: If you have only one disk drive you should enter 1 in answer to DUPLICATE SLOT DEFAULT = and press RETURN. Follow screen instructions.

Copying a file from one disk to another

1 Insert disk containing file to be copied.

2 Type LOAD filename .

3 Remove disk.

4 Insert disk on which file is to be copied.

5 Type SAVE filename .

Renaming a file on disk

There may be a reason for you to rename a file which has been saved.

1 Insert selected disk in Drive 1.

2 Call up CATALOG.

3 Select name of the file you wish to rename.

4 Type RENAME oldname, newname .

Printing a file from disk

1 Insert selected disk in Drive 1.

2 Call up CATALOG.

3 Select name of file to be printed.

4 If you want to print a listing of your program:
Type LOAD filename .

5 *Check that the printer has a supply of paper.*

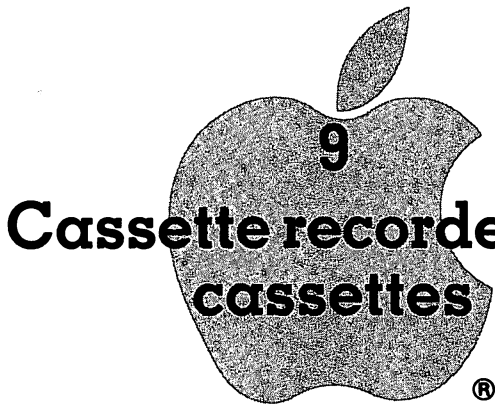
6 Turn printer on.

7 Type PR#1 (to deflect output from screen to printer).

8 Type LIST to obtain listing of program or RUN to obtain the results of the program.

9 When printing finished:
Type PR#0 (to deflect output back to screen).

10 Turn printer off.



9 Cassette recorder and cassettes

Omit this chapter if you are using only a disk drive and disks.

Starting the system

1 Turn machine on. Call prompt and cursor.
2 Clear screen. HOME or ESC @ (SHIFT–2)
3 Insert cassette (use Color Demosoft) into cassette recorder.
4 Rewind tape to beginning.
5 Set volume.
6 Depress PLAY button.

Loading the cassette

1 Type LOAD and press RETURN. Cursor will disappear.

2 One of the following will happen:

a SYNTAX ERROR	Stop cassette Rewind cassette to beginning Increase volume Depress PLAY button Type HOME Type LOAD and press RETURN
b Error message or nothing	Depress CONTROL–RESET then CONTROL–B and RETURN. Follow instructions in (a)

c Computer will beep	Either an error message will appear — follow instructions in (b) — or prompt and cursor will appear, to indicate that contents of tape are loaded into computer memory (RAM).

3 Mark the volume setting on the cassette recorder for future use.

4 Stop cassette recorder.

5 Rewind tape to beginning and remove cassette.

Running the program

Now that the contents of the tape are safely stored in the computer memory:

1 Type RUN (and press RETURN)

2 Menu (list of contents) is displayed.

3 Follow instructions on screen.

4 Depress RETURN to stop demonstration and return to menu.

You can then use any other prerecorded cassette in the same way. Some of the prerecorded programs available are Color Demosoft, Phone List, Brian's Theme, Lemonade, Penny Arcade, and Little Brick Out (you would have to use the paddles for this last one). Refer to chapter 8 for methods of using some of these programs.

Saving programs on cassette

When you have entered a program into computer memory, and have trialled that program, you may wish to save it on cassette.

1 Put blank cassette tape in cassette recorder.

2 Rewind tape to beginning, then advance leader tape and set counter to 0.

3 Depress RECORD and PLAY buttons. Immediately type SAVE .

4 Computer will beep to signal start of recording, and will beep again to signal end of recording. Remember that you may need to adjust the volume.

5 Depress STOP button on recorder after second beep.

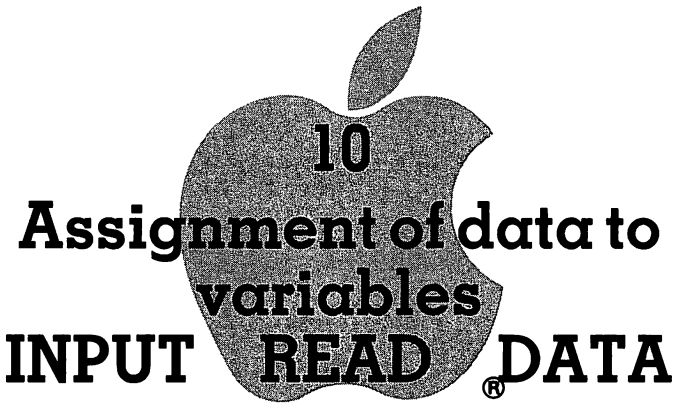
6 Make a note of counter number.

7 Type NEW and LIST to check that program has gone from computer memory.

8 Follow instructions for loading the cassette to make sure that the program has been saved.

9 Keep a record of programs saved on each cassette, with counter starting and finishing numbers.

10 Provided that you keep a record as suggested above, you can save further programs on the cassette by commencing the next save just after the counter number which indicated the finish of the previous program.



Now that you know how to save your programs on disk or cassette, you should save each program, selecting an appropriate filename.

Assigning data to variables

In previous programs you assigned:

- numeric data to numeric variables,

  eg LET A = 24 or A = 24
  LET B = 76 or B = 76

- string data to string variables,

  eg LET A$ = "NAME" or A$ = "NAME"
  LET C$ = "COST" or C$ = "COST"

Data can be also be assigned by using:

INPUT, READ and DATA

The same rules apply for assigning data by these two methods,

ie numeric data must use a numeric variable, string data must use a string variable.

INPUT statement

The INPUT statement allows data to be entered at the keyboard while the program is running. This means that you, as the operator, can enter data as required.

The RUN of the program is stopped to allow data entry. The RUN will not continue until the operator supplies the requested data.

Enter

NEW

```
10  PRINT "ENTER YOUR NAME"
20  INPUT N$
30  INPUT "ENTER TODAY'S DATE DD/MM/YY ";D$
40  PRINT "ENTER A NUMBER 1-10"
50  INPUT N1
60  INPUT "ENTER A NUMBER 50-100";N2
70  PRINT "HELLO ";N$: PRINT
80  PRINT "THE DATE IS ";D$: PRINT
90  PRINT "YOUR NUMBERS ARE ";N1;" AND ";N2
9999  END
```

The INPUT statement can be used as in:

- statements 10 and 20 — when the program is RUN, a question mark appears on the screen after the prompt message "Enter your name". You then enter the requested data. Quotation marks are not required around the input entries.
- statement 30—when the program is RUN, the prompt message "Enter today's date DD/MM/YY" appears on the screen, without a following question mark. You then enter the requested data. Quotation marks are not required around the input entries.

LIST and RUN

Enter this program:

NEW

```
10  REM   USING INPUT STATEMENT
20  PRINT "ENTER YOUR NAME"
30  INPUT N$
40  INPUT "ENTER TODAY'S DATE DD/MM/YY ";D$
50  PRINT "ENTER TWO PRICES UNDER $10"
60  INPUT P1,P2
70  INPUT "ENTER TWO NUMBERS UNDER 20 ";N1,N2
80  PRINT : PRINT
90  PRINT "EXERCISE NO. 8": PRINT
100  PRINT "INVOICE TO:",N$: PRINT
110  PRINT "AS AT",D$: PRINT
120  PRINT "QUANTITY","PRICE","COST": PRINT
130  PRINT N1,P1,N1 * P1: PRINT
140  PRINT N2,P2,N2 * P2: PRINT
150  PRINT "TOTAL COST",,N1 * P1 + N2 * P2
9999  END
```

LIST

RUN—remember that when the prompt message appears on the screen, you must enter the requested data. If more than one entry is requested (eg statements 50 and 70) separate the entries with a comma.

The advantage of this program which uses the INPUT statement is that the program will not need alteration to insert a different name, date or other details. These are supplied by the operator while the program is running.

Exercise no 9

Refer to exercise no 7 in chapter 7.

Write a program, using INPUT statements, which could be used to find the area of any rectangle.

Exercise no 10

Using INPUT statements, write a program which will print the square root of any number entered.

READ and DATA statements

Another way to assign data is to use READ and DATA statements. Two separate statements must be used. The READ statement assigns the variables to the data items, while the DATA statement supplies the data.

Enter

NEW

```
10  READ A,B$
20  PRINT A;B$
30  DATA  250," DOLLARS"
9999  END
```

LIST and RUN

When this program is run, the system automatically assigns the variables to the data—the first variable (A) to the first data item (250), the second variable (B$) to the second data item (DOLLARS).

It is essential that the type of variable (numeric or string) in the READ statement matches the type of data (numeric or string) in the DATA statement.

1 Enter this program to assign numeric data (15, 32, 56) to numeric variables (A,B,C) by means of READ and DATA statements.

NEW

```
10  REM  ASSIGNMENT OF NUMERIC DATA
20  PRINT "EXERCISE NO. 11"
30  PRINT "***************": PRINT
40  READ A,B,C
50  PRINT A,B,C
60  PRINT : PRINT
70  PRINT A,B,C
80  DATA  15, 32, 56
9999  END
```

LIST and RUN

What was the effect of statement no 60? (Printed two blank lines).

In RUNning this program, the computer assigned the value of the first data item (15) to the first variable (A); the second data item (32) to the second variable (B); and the third data item (56) to the third variable (C).

2 Now: Change exercise 11 to exercise 12.
Insert a statement to print your name.
Change statement 60 so that one blank line will be printed.
Insert a statement to instruct that D = A + B + C.
Insert a statement to print that "A + B + C =";D
LIST and RUN

3 Refer to the program for exercise no 4 in chapter 7.
Work out how to change that program so that the values of N and P are assigned by READ and DATA statements.
Enter your program as exercise no 13.

Remember: NEW LIST RUN

4 Refer to exercise no 7 in chapter 7.
Write a program to solve that problem, using READ and DATA statements to assign the variables L (length) and W(width).
Enter your program as exercise no 14.

Remember: NEW LIST RUN

5 In the following exercise, string data is assigned to string variables, using the READ and DATA statements.
Enter this program:

```
NEW
10  REM  ASSIGNMENT OF STRING DATA
20  PRINT "EXERCISE NO. 15": PRINT
30  READ H$,N$,D$
40  READ A$,J$,P$,S$
50  READ W$,X$,Y$,Z$
60  PRINT  TAB( 6);H$: PRINT : PRINT
70  PRINT N$,D$: PRINT
80  PRINT A$,W$: PRINT J$,X$
90  PRINT P$,Y$: PRINT S$,Z$
100  PRINT : PRINT
110  PRINT  TAB( 6);H$: PRINT
120  PRINT D$,N$: PRINT : PRINT
130  PRINT X$,J$: PRINT Z$,S$
140  PRINT W$,A$: PRINT Y$,P$
150  DATA  "BIRTHDAY LIST","NAME","DATE"
160  DATA   "ALAN","JUDITH","PETER","SUSAN"
170  DATA   "17 JULY","30 JANUARY"
180  DATA   "12 OCTOBER","14 APRIL"
9999  END
```

LIST and RUN

Remember: the first string data item (BIRTHDAY LIST) is assigned to the first string variable (H$), the second string data item (NAME) to the second string variable (N$), and so on.

Once the data has been assigned to variables, those variables may be used in any order and—if required—more than once. Notice that your program used the same data to produce two lists, one in alphabetical order by name, the other in chronological (date) order.

6 Write a program which will print a list of the titles of three songs, and the names of the singers for each song. Use READ and DATA statements to assign the variables. Provide a heading and column headings.

Enter your program as exercise no 16.

Remember: NEW LIST RUN

7 Write a program (exercise no 17) which will print the following table:

CAPITAL CITIES

STATE	CAPITAL
Queensland	Brisbane
Victoria	Melbourne
Western Australia	Perth

Use READ and DATA statements to assign the variables, other than the heading and column headings.

Remember: NEW LIST RUN

8 Write a program which will print the names of four suburbs (string data) and the postcodes for those suburbs (numeric data). Use READ and DATA statements to assign variables. Include a heading and column headings for the output.

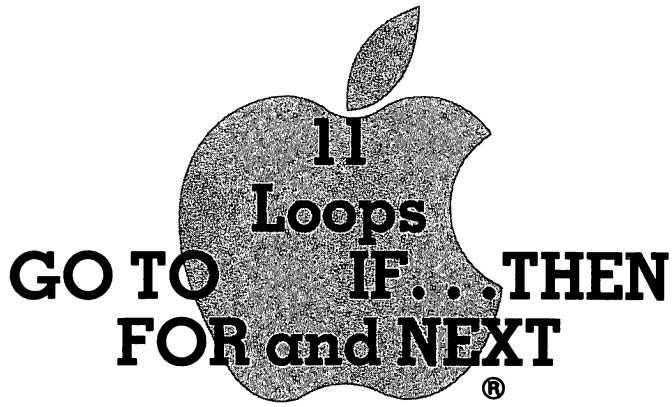
Enter your program as exercise no 18.

9 Write a program to produce the following output. Use READ and DATA statements to assign variables, other than headings.

STUDENT RESULTS

NAME	SUBJECT	PER CENT
Alicia	English	76
Michael	Science	81
Heidi	English	69
Nicholas	Maths	73

Enter your program as exercise no 19.



When the solution to a problem requires that some instructions should be repeated, a *loop* may be used. This simply means that when the program RUN reaches a particular statement, it will automatically return to a previous statement and repeat the instructions given.

GO TO statement

The GO TO statement instructs the computer to GO TO another statement number. In the following program, statement 70 instructs the computer to go back to statement 50 and repeat the instructions. This is called a **loop.**

The GO TO statement is called an **unconditional** statement because the computer must obey.

1 Enter this program:

```
NEW

10  REM  CALCULATING THE PRODUCT OF TWO VALUES
20  REM  N=VALUE 1, P = VALUE 2
30  PRINT "EXERCISE NO. 20": PRINT
40  PRINT "VALUE 1","VALUE 2","PRODUCT": PRINT
50  READ N,P
60  PRINT N,P,N * P
70  GOTO 50
80  DATA  10, 20, 15, 7, 23, 6, 4, 81
9999  END
```

LIST and RUN

After the program is run, notice the error message OUT OF DATA ERROR IN 50 . The computer followed your instructions until all the data (statement 80) had been used. Then, following your instruction in statement 70, the computer went back to statement 50, but could not find any more data to read. The error message explains to you why the computer could not follow your instructions any further.

2 Design a short program, using the GO TO statement, to find the price of each article.

14 articles	total cost $310.66
27 articles	total cost $866.97
21 articles	total cost $176.82

Call your program exercise no 21. Select your own column headings. Price will equal Cost divided by Number.

Remember: NEW—unless you can adapt exercise 20
LIST
RUN

3 Here is another example of a GO TO statement. Remember that GO TO means that the computer must obey the command—it is an **unconditional** statement.

In this program the GO TO statement will cause a never-ending loop, so it will be up to you to stop the RUN execution when you have seen how it works.

To stop the run execution, use CTRL–C (ie press the two keys CTRL and C together).

Enter this program:

```
NEW
10  REM  GO TO AND CONTINUOUS LOOP
20  PRINT "EXERCISE NO. 22": PRINT
30  PRINT "YOUR NAME": PRINT
40  READ L
50  PRINT L
60  L = L + 2
70  GOTO 50
80  DATA  2
9999  END
```

Look at this program and try to work out why the GO TO statement will cause a never ending loop.

LIST

RUN

Remember: Stop the RUN with CTRL–C,
Resume the RUN with CONT.

Notes:

a When you stopped the program RUN, you received a message BREAK IN (no.) . This means that when you depressed

CTRL–C the RUN of the program was stopped at statement (no.).

b What does statement 60 mean?

L = L + 2 is actually an instruction to the computer:

'Let the value of L be replaced by the value of L + 2.'

c If you want to RUN the program again, type CONT (for continue), but remember to use CTRL–C to stop the RUN.

IF . . . THEN statement

The IF . . . THEN statement is a **conditional** statement, because it instructs the computer:

IF (a certain condition is met) THEN (go to another statement).

Remember the continuous loop in exercise 22? Now you are going to insert a condition that will end that loop.

1 Enter this program:

```
NEW

10  REM  IF...THEN STATEMENT
20  PRINT "EXERCISE NO. 23": PRINT
30  PRINT "YOUR NAME": PRINT
40  READ L
50  PRINT L
60  L = L + 2
70  IF L > 20 THEN 9999
80  GOTO 50
90  DATA  2
9999  END
```

The sign > means 'greater than', so statement 70 instructs the computer:

IF L is greater than 20, THEN go to statement 9999.

LIST and RUN

2 Amend exercise 23 by:

Changing statement 50 to PRINT L, (What effect will the comma have?)
Changing statement 60 to L= L−2
Changing statement 70 to IF L<0 then 9999
Changing statement 90 to DATA 100

The sign < means 'less than' so statement 70 will now instruct:

IF L is less than 0, THEN go to statement 9999.

LIST and RUN

3 Look carefully at the following program and try to understand it before you enter it.

NEW

```
10  REM  IF...THEN STATEMENT
20  REM  P = PRICE, R = RATE
30  PRINT "EXERCISE NO. 24": PRINT
40  PRINT "YOUR NAME": PRINT
50  PRINT  TAB( 10);"SALES TAX SCHEDULE": PRINT
60  PRINT "PRICE","RATE %","TAX": PRINT
70  READ P,R
80  PRINT P,R,P * R
90  P = P + 5
100  IF P = 105 THEN 9999
110  GOTO 80
120  DATA  20, .10
9999  END
```

Have you worked out how this program will RUN?

Enter, LIST and RUN the program.

4 Design a short program (exercise 25) to produce the following output. Use READ and DATA, GO TO, IF . . . THEN statements.

SCHEDULE OF CASUAL RATES

Hours Worked	Hourly Rate	Wages Due
2	6.80	(Hours * Rate)
4	6.80	
6	6.80	
8	6.80	
up to 40		

5 Enter this program, which will allow you to INPUT two numbers, PRINT both numbers, and determine which is the larger number.

NEW

```
10  REM  DETERMINING LARGER NUMBER
20  REM  N1=NUMBER, N2=NUMBER
30  PRINT "EXERCISE NO. 26": PRINT
40  PRINT "YOUR NAME": PRINT
50  INPUT "ENTER TWO NUMBERS ";N1,N2: PRINT
60  PRINT "NUMBERS ARE: ",N1,N2: PRINT
70  IF N1 > N2 THEN 100
80  PRINT "LARGER NUMBER IS:",N2
90  GOTO 9999
100  PRINT "LARGER NUMBER IS:",N1
9999  END
```

LIST and RUN

6 Write a program (exercise no 27) using INPUT, IF . . . THEN and GO TO statements, which will:

a INPUT a student's name

b INPUT exam marks for that student for three exams where the maximum marks for each exam are 25, 35 and 40

c calculate the total marks

d print student name, total marks, and
PASS (if total marks are 50 or above)
FAIL (if total marks are less than 50).

Supply a suitable heading and column headings for your output.

FOR and NEXT statements

The number of times an instruction is performed can be controlled by FOR and NEXT statements, which cause the program to loop.

In the following example, the instructions between statements 60 and 90 will be performed ten (10) times.

FOR sets the counter (C) to start at a selected number (eg 1) and to finish at a selected number (eg 10).

STEP sets the seclected number by which the counter (C) will be increased each time the instruction is performed (eg STEP 1 means increase the value of the counter by +1).

NEXT increases the counter (C) by the step shown (+1). If the counter is less than the selected finishing number (eg 10), the statements in the loop are repeated. If the counter is equal to the selected finishing number (eg 10), the loop is completed. The program would then continue from the statement following the NEXT statement.

1 Study this program.
NEW

```
10  REM   FOR AND NEXT STATEMENTS
20  REM  C = COUNTER FOR LOOP
30  PRINT "EXERCISE NO. 28": PRINT
40  PRINT "YOUR NAME": PRINT
50  READ A,B
60  FOR C = 1 TO 10 STEP 1
70  PRINT A,B
80  A = A + B
90  NEXT C
100  DATA  10, 2
9999  END
```

Statement 60 (FOR C=1 TO 10 STEP 1) is the starting point for the loop. The counter C will start at 1.

Statement 90 (NEXT C) is the instruction to go to statement 60 and increase the counter by the STEP shown, (in this program step 1). So the counter becomes 2.

The program loops (works) from statement 60 to statement 90 until the counter = 10. The loop will then END.

Enter, LIST and RUN this program.

2 Now make these changes:

Change to exercise 29
Change the FOR statement to step 2.
Change A = A + B to A = A * B

LIST and RUN

3 It is also possible to have the loop counter decrease in value by using a minus step. In this case, the selected starting point for the loop must be larger than the selected finishing number,

eg FOR C = 100 TO 20 STEP −5

The counter (C) will start at 100 and decrease by 5 each time the loop is executed. When the counter is equal to 20, the loop is stopped.

NEW

```
10  REM  USING STEP MINUS
20  PRINT "EXERCISE NO. 30": PRINT
30  FOR C = 100 TO 20 STEP  - 5
40  PRINT C
50  NEXT C
60  PRINT
70  PRINT "EXAMPLE OF MINUS STEP"
9999  END
```

LIST and RUN

Notice that, in this example, the actual decreasing value of the counter has been printed.

4 What do you think will happen when you RUN this program?

NEW

```
10  REM  SIXES AND SEVENS
20  REM  C = COUNTER
30  PRINT "EXERCISE NO. 31": PRINT
40  PRINT "YOUR NAME": PRINT
50  READ A,B
60  FOR C = 200 TO 0 STEP  - 1
70  PRINT A;B;
80  NEXT C
90  DATA  6, 7
9999  END
```

Enter, LIST and RUN this program.

5 In this example, the READ and PRINT statements are contained within the loop. The first time through the loop, the first two DATA items are read. The second time through the loop, the third and fourth data items are read, and so on.

Enter
NEW

```
10  REM  READ INSIDE LOOP
20  REM  C = COUNTER
30  REM  S$=STATE, C$=CAPITAL
40  PRINT "EXERCISE NO. 32": PRINT
50  PRINT "CAPITAL CITIES": PRINT
60  PRINT "STATE","CITY": PRINT
70  FOR C = 1 TO 3 STEP 1
80  READ S$,C$
90  PRINT S$,C$
100  NEXT C
110  DATA   "TASMANIA","HOBART"
120  DATA  "S.AUSTRALIA","ADELAIDE"
130  DATA   "N.S.W.","SYDNEY"
9999  END
```

LIST and RUN

Did you notice that this program produces the same output as exercise no 17 in chapter 8? By using the loop only two variables were used, and only one **PRINT** statement.

6 Write a program (exercise no 33) using **READ** and **DATA** statements, and a **FOR** and **NEXT** loop, to produce the same output as exercise no 18 in chapter 8. Use only two variables and contain the **READ** and **PRINT** statements within the loop.

7 Refer to the output for exercise 19 in chapter 8. Use **READ** and **DATA**, and **FOR** and **NEXT** statements, to produce that output. Use three variables only and contain the **READ** and **PRINT** statements inside the loop.

Enter your program as exercise no 34.

8 Write a program which will print a list of the names, suburbs and phone numbers of six people.

Enter your program as exercise no 35.

CTRL–C AND CONT

Remember: If you want to stop the RUN of any program:
CTRL–C to stop RUN; CONT to continue RUN.

CTRL–S

If you want to stop the LISTing of any program:
CTRL–S to stop LIST; CTRL–S to continue LIST.

Using the printer

Before proceeding check that the printer has a supply of paper inserted. See the manual for your particular printer for details of paper insertion.

The interface card for the printer is normally in slot 1. To deflect output from the screen to the printer, you would type PR#1 (where # = international symbol for number, and 1 = slot number). If your computer has the interface card in a different slot, simply substitute that slot number.

1 Turn printer on.

2 Type PR#1 (RETURN) to deflect output from screen to printer.

The following commands will not display on the screen.

3 Type CTRL–I80N (RETURN) to set printer to 80 spaces wide.

4 Type LIST (RETURN) or RUN (RETURN).

5 When printing has finished, type PR#0 (RETURN) to deflect output back to screen.

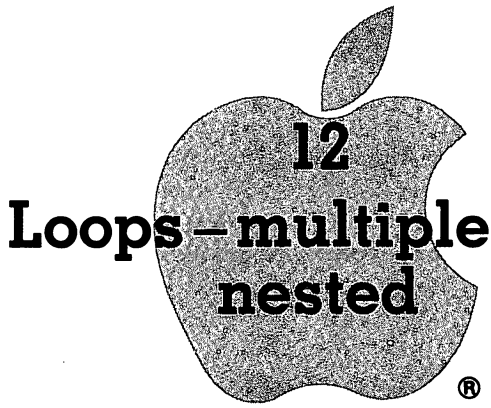
6 Turn printer off.

Re-enter the program for exercise 15.

LIST and RUN on the screen.
LIST and RUN on the printer.

Re-enter the program for exercise 16.

LIST and RUN the program on the printer.



12 Loops – multiple and nested

A program may use more than one loop to repeat instructions. The loops may be separate parts of the program, or they may be nested (ie one inside another).

Multiple loops

1 Enter
NEW

```
10  FOR C = 1 TO 3 STEP 1
20  PRINT C
30  NEXT C
40  PRINT "END OF FIRST LOOP": PRINT
50  FOR K = 2 TO 10 STEP 2
60  PRINT K
70  NEXT K
80  PRINT "END OF SECOND LOOP"
9999  END
```

LIST and RUN

2 A transport company needs to calculate the number of trips to move five quantities of soil (1890 kilograms per trip) at the cost of 35c per kilogram. Also required is a separate table showing earnings for six employees, based on an hourly rate of $11.70. This program uses two loops to solve their problem. The data statements contain the five quantities of soil and the hours worked by six employees.

Enter
NEW

```
10  REM  USING TWO LOOPS
20  REM  Q = QUANTITY, C = COST, H = HOURS
```

(continued)

```
30  PRINT "EXERCISE NO. 36": PRINT
40  PRINT "QUANTITY","TRIPS","COST": PRINT
50  FOR C = 1 TO 5 STEP 1
60  READ Q
70  PRINT Q,Q / 1890,Q * .35
80  NEXT C
90  PRINT
100  PRINT "HOURS","EARNINGS": PRINT
110  FOR K = 1 TO 6 STEP 1
120  READ H
130  PRINT H,H * 11.70
140  NEXT K
150  DATA  13230, 21735, 4347, 11340, 18711
160  DATA  27, 14, 31, 8, 40, 21
9999  END
```

LIST and RUN

3 Write a program (exercise no 37) which will

a convert five Celcius temperatures to Fahrenheit temperatures.
$F = (C/5)9+32$

b convert five Fahrenheit temperatures to Celsius temperatures.
$C = (F-32)5/9$

Use two loops, and print two separate tables with suitable headings.

Nested loops

1 The term **nested loops** describes two loops, one of which is totally enclosed within the other.

Enter

NEW

```
10  FOR C = 1 TO 3 STEP 1
20  PRINT "LOOP ";C: PRINT
30  FOR K = 1 TO 4 STEP 1
40  PRINT K
50  NEXT K
60  PRINT
70  NEXT C
9999  END
```

LIST and RUN

Notice that the FOR and NEXT statements for counter K (statements 30 and 50) are totally enclosed within the FOR and NEXT statements for counter C (statements 10 and 70).

2 In this program, a payroll is being calculated for four employees, based on daily hours worked over a period of five days. Hourly rate is $8.75.

Hours over 35 are paid at time and a half.

Using nested loops, the first loop reads and prints the employee name. The second loop reads the hours, adds the hours, calculates and prints the earnings.

Notice that total hours (T) is set to zero each time a name is read.

Enter, LIST and RUN this program.

Enter

NEW

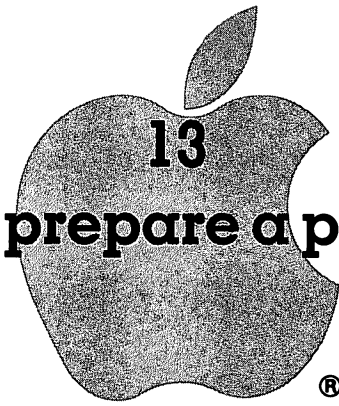
```
10   REM  CALCULATING PAYROLL
20   REM  N$=NAME,H=HOURS,T=TOTAL HOURS
30   PRINT "EXERCISE NO. 38": PRINT
40   PRINT  TAB( 16);"PAYROLL": PRINT
50   PRINT "NAME","HOURS","EARNINGS": PRINT
60   FOR C = 1 TO 4 STEP 1
70   T = 0
80   READ N$
90   PRINT N$,
100   FOR K = 1 TO 5 STEP 1
110   READ H
120   T = T + H
130   NEXT K
140   IF T > 35 THEN 170
150   PRINT T,T * 8.75
160   GOTO 180
170   PRINT T,(35 * 8.75) + ((T - 35) * (8.75 * 1.5))
180   PRINT
190   NEXT C
200   DATA  "SLOAN", 7, 6.5, 8, 6, 4
210   DATA  "HUNTER", 7, 8, 6, 7, 7
220   DATA  "MAY", 8, 7.5, 8, 8.5, 6
230   DATA  "RILEY", 4, 6, 5, 9, 4
9999   END
```

3 Refer to exercise 27 in chapter 9. Write a program (exercise no 39) which will READ the data for five students (name and three results) and print a table of name, total marks, and Pass or Fail. You should use two loops. Provide suitable heading and column headings.

4 Write a program (exercise no 40) which will print a table showing the credit terms offered by a store on each of five prices. Your output should be:

CREDIT TERMS

PRICE	DEPOSIT	SIX	TWELVE
$	10%	MONTHS	MONTHS
(P)	(D)		
	(P*.10)	(P−D)/6	(P−D)/12



13 How to prepare a program

So far, the programs you have used have all been very simple. If you plan to develop your own programs, you will need to learn how to solve the problem and then transfer this input to your program.

To solve any problem, you should first divide it into smaller sections to make the solution easier.

Steps to follow

1 Consider the problem carefully.
2 Select the format you require for your answer.
3 Identify the data you have been given.
4 Assign variables as required.
5 Decide what calculations will be needed.
6 Decide what decisions need to be made.
7 Prepare guidelines.
8 Write program.

Now, put these steps into action with a simple exercise.
Turn back to exercise 3 in chapter 7. The question for that exercise could have stated:
Write a program which will print:

EXERCISE NO. 3
****

Your name
****

A = 50
B = 10
C = 12

A+B*C = (A + B * C)

A/B*C = (A/B*C)

Step 1	
Consider problem	Read question carefully
Step 2	
Select layout	Already available
Step 3	
Identify given data	50 10 12
Step 4	
Assign variables	A B C
Step 5	
What calculations	A + B * C A/B * C
Step 6	
What decisions?	None in this example
Step 7	
Prepare guidelines	Use a flowchart, *or* List as a series of short sentences
Step 8	
Write program	From your guidelines

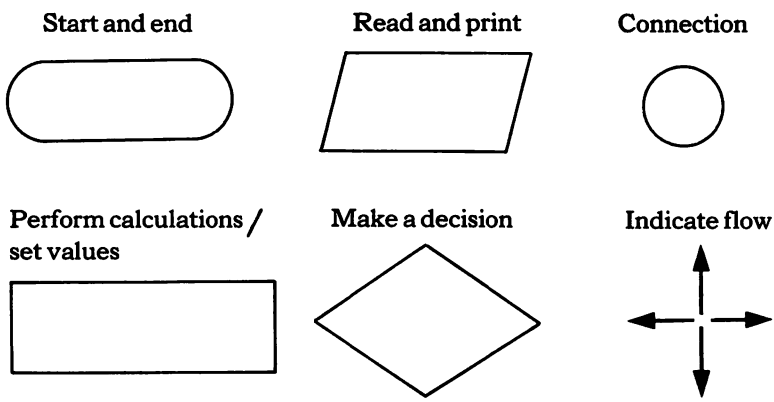
Guidelines

In preparing guidelines, you may use either:

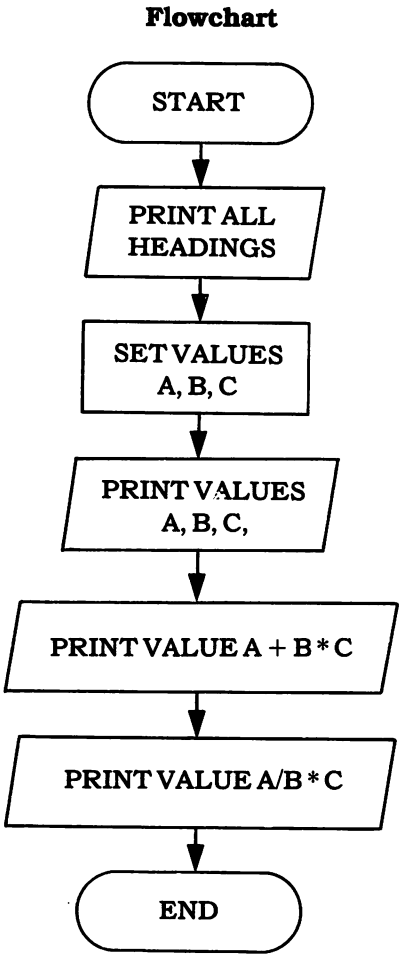
- a flowchart — a diagram of the logical steps needed, *or*
- short sentences — to outline each step

For the exercises in this chapter, use both methods and then decide which approach you prefer.

When using a flowchart, the following shapes are often used to represent different steps.



Your guidelines for exercise 3 could be:



Note: REM statements are not shown in either flowchart or sentences. They are used to identify and clarify your program.

1 By following the examples, you should now be able to write guidelines for exercise 4 from this question.

Write a program which will print:

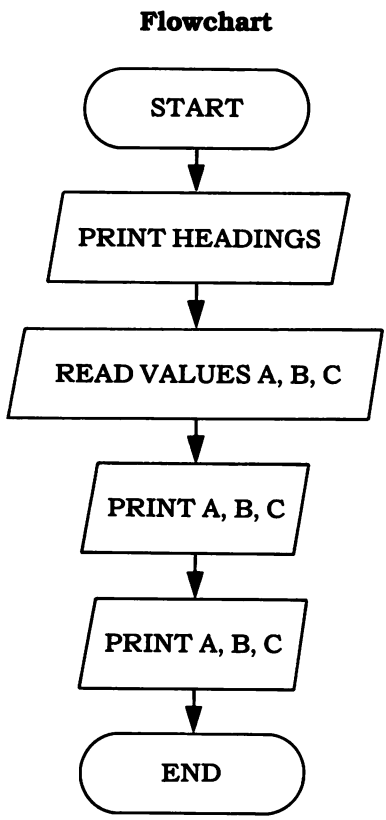
EXERCISE NO. 4
****

Your name
****

NUMBER	PRICE	COST
48	17.58	(Number × price)

2 Now prepare guidelines for exercises 5, 6 and 7 from chapter 7.

3 Here are the guidelines for exercise 11 from chapter 10:



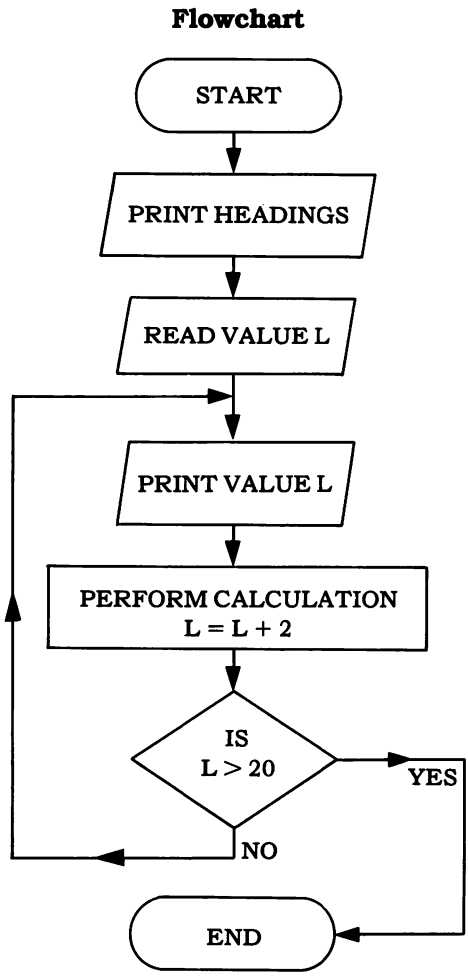
Note: DATA statements are not shown in either flowchart or sentences.

4 Prepare guidelines for exercise 13 from chapter 10.

5 This question refers to exercise 23 from chapter 11.
Write a program which will print:

EXERCISE NO. 23
YOUR NAME
2
4
6
up to
20

These are the guidelines:



Note: This program will loop until the value of L is greater than 20.

6 Prepare guidelines for exercises 24 and 25 from chapter 11.

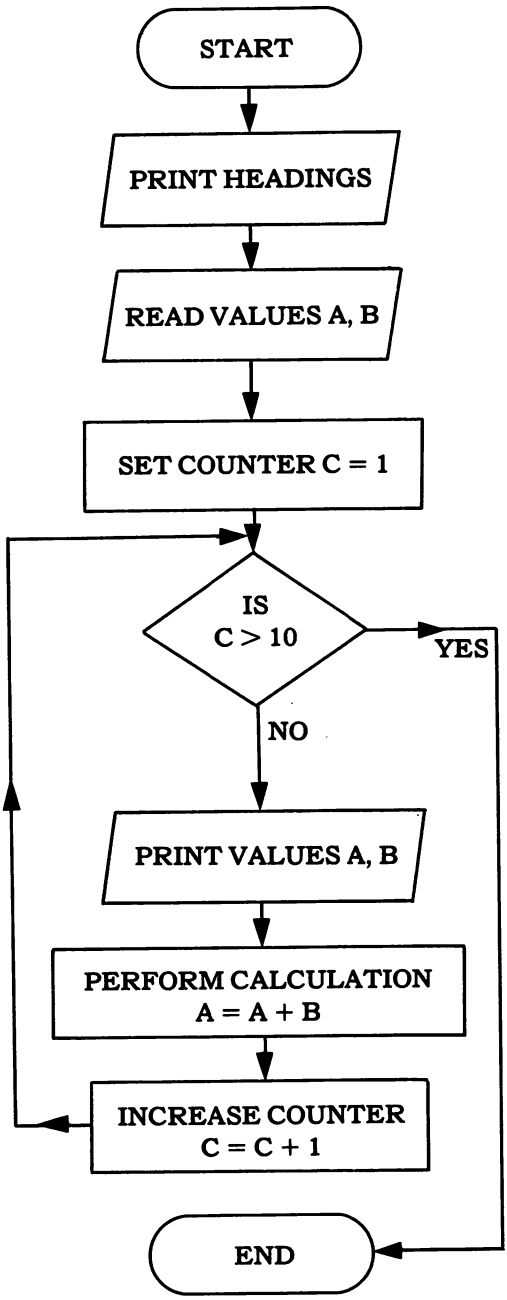
7 The question for exercise 28 from chapter 11 could state:
Write a program which will print:

EXERCISE 28
Your name

10	2
12	2
14	2
16	2
up to	
28	2

Here are the guidelines:

Flowchart



Sentences

Start program

Print headings

Read data for A and B

Set counter initially to 1 so that the number of loops may be counted

Is counter greater than 10?

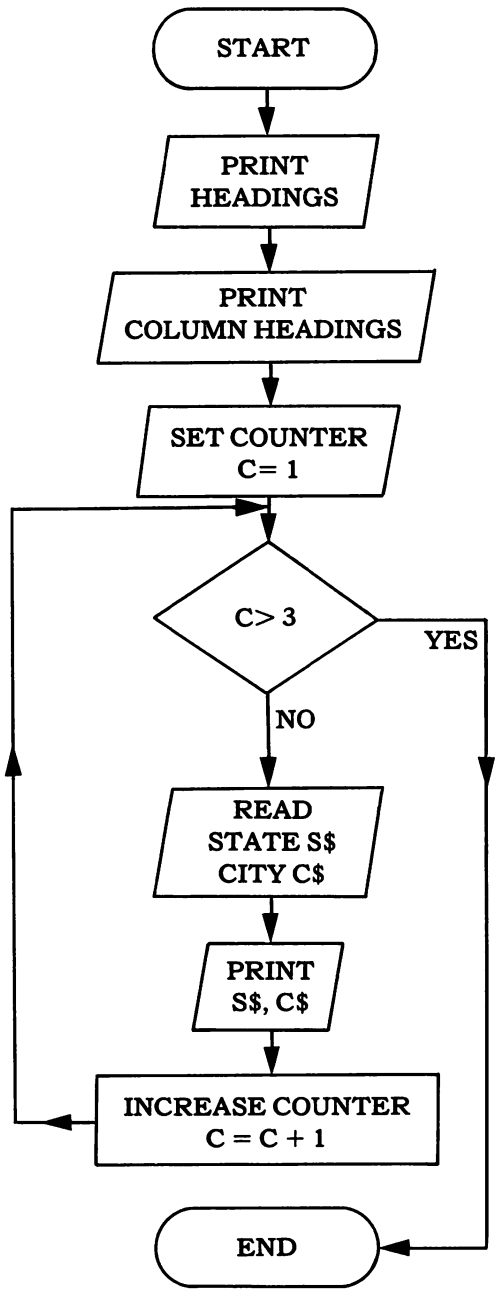
If NO—print A, B
increase A by B
increase counter by 1
repeat loop

If YES—end of loop

Note: The program will loop until the value of C is greater than 10.

8 Here are the guidelines for exercise 32 from chapter 11.

Flowchart



Sentences

Start

Print headings

Print column headings

Set counter initially to 1 so that number of loops may be counted

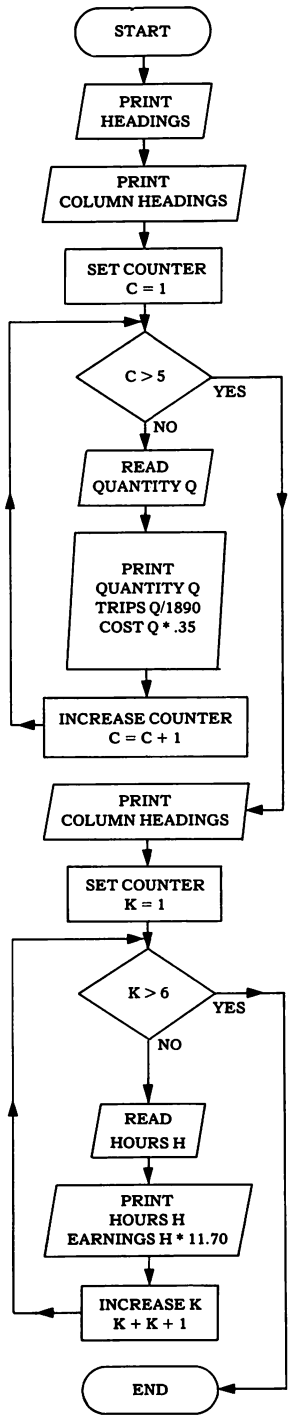
Is counter greater than 3?

If NO—read S$, C$
print S$, C$
increase counter by 1
repeat loop

If YES—end of loop

9 These guidelines apply to exercise 36 from chapter 12:

Flowchart



Sentences

Start

Print headings

Print column headings

Set Counter C to 1

Is C greater than 5?

If NO—read quantity Q
print quantity Q
print trips Q/1890
print cost Q × .35
increase counter by 1
repeat loop

If YES—next section of program

Print column headings

Set counter K to 1

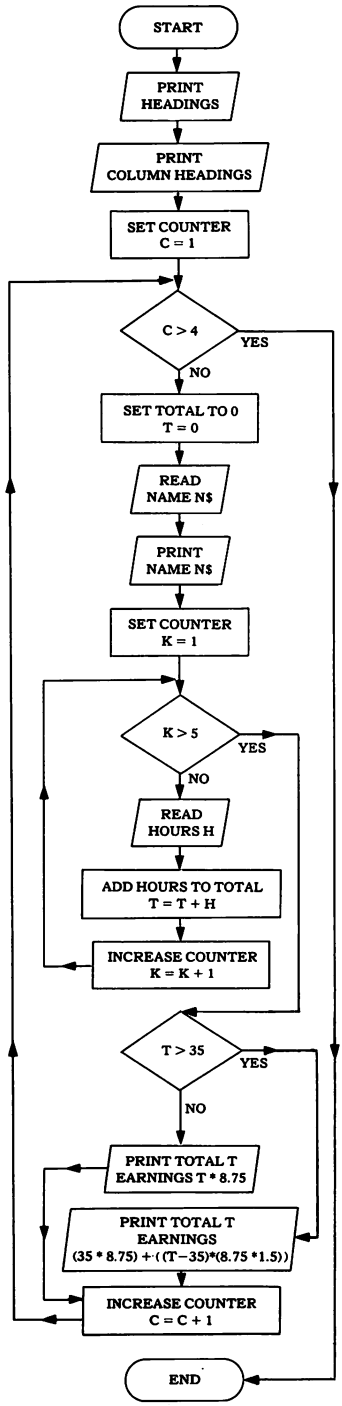
Is K greater than 6?

If NO—read hours H
print hours H
print earnings H × 11.70
increase counter by 1
repeat loop

If YES—end of loop

10 Exercise 38 from chapter 12 could have the following guidelines:

Flowchart



Sentences

Start

Print headings

Print column headings

Set Counter C to 1

Is C > 4?

If NO—set total T to 0
read name N$
print name N$
set counter K to 1
is K > 5?

If NO—read hours H
add hours to total T
increase counter K by 1
repeat loop (K)

If YES—T > 35?

If NO—print total
print earnings
increase C by 1
repeat loop (C)

If YES—print total
print earnings
include overtime
increase C by 1
repeat loop (C)

If YES—end of loop

11 Follow the steps to solve this problem:

Exercise no 41
Prepare guidelines
Write program
Enter program
RUN program

You have purchased a car for $8750. You paid a deposit of $2030 and obtained a loan for the balance over a term of 24 months. You have agreed to make a monthly payment of $280.

Your answer should show *all* details, and should include a table showing month (1 to 24), payment and balance (after payment).

Some hints to help:

- Use a counter to indicate the months 1 to 24.
- The value of the counter can be printed.
- First balance will equal cost minus deposit.
- Each following balance will equal balance minus payment.

Good programming!

Note: Programs for questions may vary in the order of steps. The only way to test whether a program is written correctly is to RUN it. If an accurate result is obtained, in the correct format, then the program is correct.

14 More statements

This chapter introduces a number of additional statements and the work involved is slightly more difficult.

Work the following examples to gain an understanding of the statements.

LIST and RUN each program.

RESTORE statement

The RESTORE statement instructs the computer to read DATA again, from the first DATA statement.

Enter

NEW

```
10  READ A,B
20  PRINT A,B
30  READ A,B
40  PRINT A,B
50  RESTORE
60  READ A,B
70  PRINT A,B
80  DATA  67, 102, 48, 76
9999  END
```

(Try this without the RESTORE statement!)

Subroutine

The instruction GOSUB directs the computer to go to the **subroutine** section of the program, which is identified by the statement number following GOSUB. The subroutine section is then executed and the final

statement in the subroutine is RETURN, which directs the computer to return to the statement following the GOSUB statement.

Enter

```
NEW
10  GOSUB 50      (GO to start of SUBroutine 50)
20  PRINT 2,4,6
30  PRINT
40  GOTO 9999
50  PRINT 1,3,5
60  RETURN        (start of subroutine)
9999  END         (RETURN to statement after GOSUB)
```

Dummy data

Dummy data is data which is included to signify the end of DATA. This is useful when the amount of data is subject to change.

Enter

```
NEW
10  READ A
20  IF A =  - 111 THEN 9999
30  PRINT A,
40  GOTO 10
50  DATA  15, 21, 36, 18, 12, -111
9999  END
```

LEN (length) statement

The LEN statement determines the number of characters in a variable.

Enter

```
NEW
10  A$ = "AUSTRALIA"
20  B$ = "APPLE MICROCOMPUTERS"
30  A =  LEN (A$):B =  LEN (B$)
40  PRINT A$,A
50  PRINT B$,B
9999  END
```

ON . . . GO TO statement

The ON . . . GO TO statement allows a program to branch to one of a number of other statements, depending on the input. The variable used for the input must be followed by a percentage sign (%).

Enter
NEW

```
10  INPUT "GUESS A NUMBER BETWEEN 1 AND 5 ";A%
20  IF A% < 1 OR A% > 5 THEN 90
30  ON A% GOTO 40,50,60,70,80
40  PRINT "YOU GUESSED NUMBER 1": GOTO 100
50  PRINT "YOU GUESSED NUMBER 2": GOTO 100
60  PRINT "YOU GUESSED NUMBER 3": GOTO 100
70  PRINT "YOU GUESSED NUMBER 4": GOTO 100
80  PRINT "YOU GUESSED NUMBER 5": GOTO 100
90  PRINT "NUMBER OUTSIDE RANGE - TRY AGAIN ": GOTO 10
100  PRINT "DO YOU WANT TO GUESS AGAIN?"
110  INPUT "ENTER Y OR N ";G$
120  IF G$ = "Y" THEN 10
9999  END
```

ON . . . GOSUB statement

This is very similar to the ON . . . GO TO statement in that the program will branch to one of a number of subroutines depending on the input. The variable used for the input must be a numeric variable.
Enter
NEW

```
10  PRINT "1 READ      2 INPUT": PRINT
20  PRINT "3 PRINT     4 SAVE": PRINT
30  INPUT "SELECT BY NUMBER ";N: PRINT
40  IF N < 1 OR N > 4 THEN 100
50  ON N GOSUB 60,70,80,90
60  PRINT "SUBROUTINE TO READ": GOTO 9999
70  PRINT "SUBROUTINE TO INPUT": GOTO 9999
80  PRINT "SUBROUTINE TO PRINT": GOTO 9999
90  PRINT "SUBROUTINE TO SAVE": GOTO 9999
100  PRINT "INCORRECT NUMBER": PRINT : GOTO 10
9999  END
```

INT (integer) statement

The INT statement will return:

- a whole number, less than or equal to a decimal number
- the nearest whole number, by adding .5 to the number in brackets.

Enter
NEW

```
10   A = 17.6582:B = 21.1247
20   PRINT A, INT (A), INT (A + .5)
30   PRINT B, INT (B), INT (B + .5)
9999   END
```

LEFT$, MID$, RIGHT$ statements

These statements will allow you to extract selected characters from a variable.

Enter
NEW

```
10   A$ = "AUSTRALIA"
20   PRINT   LEFT$ (A$,4)      first 4 characters
30   PRINT   RIGHT$ (A$,3)     last 3 characters
40   PRINT   MID$ (A$,5,2)     starts at character 5;
50   B$ = "25/12/198-"         prints 2 characters
60   PRINT   LEFT$ (B$,2)
70   PRINT   RIGHT$ (B$,4)
80   PRINT   MID$ (B$,4,2)
9999   END
```

VAL statement

The VAL statement converts a string variable (or part of a string variable) to a numeric value.

Enter
NEW

```
10 N$ = "1234"
20 N =  VAL (N$)
30 PRINT N$,N
40 D$ = "25/12/1983"
50 D =  VAL (D$)
60 PRINT D$,D
70 Y =  VAL ( RIGHT$ (D$,2))
80 PRINT D$,Y
90 M =  VAL ( MID$ (D$,4,2))
100 PRINT D$,M
110 Z =  VAL ( LEFT$ (D$,2))
120  PRINT D$,Z
9999 END
```

Now you will progressively develop a program using, as an example, data relating to students.

Work carefully through each section, following instructions given.

The problem you have been given is to write a program which will return student information in different formats.

Notes:

1 Be sure to enter statement numbers exactly as shown, to allow for further statements to be entered as the program is developed.
2 Keep a printed copy of the program. LIST and RUN for comparisons as the program is developed.

```
10  REM  TO PROCESS STUDENT RESULTS
20  REM  VARIABLES DESCRIPTION
30  REM  DATE        D$
40  REM  NAME        N$
50  REM  ADDRESS     A$
60  REM  PHONE       P$
70  REM  BIRTHDATE   B$
110  PRINT "ENTER CURRENT DATE DD/MM/YY"
120  PRINT : INPUT D$
160  H$ = "PERSONAL DETAILS"
180  PRINT H$
185  PRINT : PRINT D$: PRINT
190  PRINT "NAME","PHONE","BIRTHDATE","ADDRESS": PRINT
200  FOR R = 1 TO 3 STEP 1
210  READ N$,A$,P$,B$
230  PRINT N$,P$,B$,A$
240  NEXT R
245  PRINT
810  DATA  "STUDENT 1","4 HILL ROAD MORNINGSIDE"
815  DATA  "396.1234","17/10/67"
830  DATA  "STUDENT 2","138 MAY ROAD CARINA"
835  DATA  "391.7222","26/08/65"
850  DATA  "STUDENT 3","21 BAKER STREET CAMP HILL"
855  DATA  "399.6214","12/10/66"
9999  END
```

SAVE as STUDENT DETAILS
LIST and RUN
PRINT LIST and RUN

Adding more data

It would be useful if your program could also provide a separate list of a student's class and the subjects in which each student is enrolled.

Enter these statements to describe the variables:

```
75   REM   CLASS        C$
80   REM   ENGLISH      E$
90   REM   MATHS        M$
100   REM   SCIENCE     S$
```

Enter these statements to include the data, which consists of the above variables in the order C$, E$, M$, S$.

(E = Enrolled N = Not enrolled)

```
820   DATA   "9B","E","E","N"
840   DATA   "9A","E","E","N"
860   DATA   "9C","E","N","E"
```

Change statement 210 to READ the extra data:

```
210   READ N$,A$,P$,B$,C$,E$,M$,S$
```

SAVE as STUDENT DETAILS

LIST and RUN

PRINT LIST and RUN

Notice that although the extra data has been READ, it has not affected the results, as no instruction was given to PRINT that extra data.

Using the RESTORE statement

In order to print another list containing students' names and subjects, the data must be READ again. This is done by using the RESTORE statement, which instructs the computer to READ data again, from the first DATA statement.

Enter these statements:

```
280   H$ = "STUDENT ENROLMENT"
300   PRINT H$
305   PRINT : PRINT D$: PRINT
310   PRINT "NAME","CLASS","ENGLISH","MATHS","SCIENCE"
315   PRINT
320   RESTORE
330   FOR R = 1 TO 3 STEP 1
340   READ N$,A$,P$,B$,C$,E$,M$,S$
360   PRINT N$,C$,E$,M$,S$
370   NEXT R
375   PRINT
590   GOTO 9999
```

SAVE as STUDENT DETAILS

LIST and RUN

PRINT LIST and RUN

Your printed result should now show two separate lists, one for Personal Details and one for Student Enrolment.

Using a subroutine

Where the same directions are to be used more than once in a program, it is a good idea to use a subroutine. This will reduce the amount of typing in a long program and ensure that the directions are performed exactly the same way every time.

In your program, two statements are already identical. They are the READ statements 210 and 340.

Enter these statements:

```
700  REM  SUBROUTINE TO READ DATA
710  READ N$,A$,P$,B$,C$,E$,M$,S$
720  RETURN
```

Change statements 210 and 340 as shown:

```
210  GOSUB 700
340  GOSUB 700
```

SAVE as STUDENT DETAILS

LIST and RUN

PRINT LIST and RUN

When the program reaches the first GOSUB 700 statement (statement 210), it is redirected to statement 700, then to statement 710 to READ the data, then to statement 720 which will RETURN the program to the statement immediately after the GOSUB statement. The program will then continue until the next GOSUB 700 statement and the process is repeated.

Using dummy data

Your program has only been designed to process the results for three students. What happens if you are asked to include details for more students? You could of course increase the counter, but you would then need to change every statement containing a counter, each time you had to include extra data.

Here's how to overcome this problem:

1 Increase all counters to a number which you know will be above the maximum required.

By increasing the counters to 100, your program will be capable of processing details for up to 100 students.

Enter:

```
200  FOR R = 1 TO 100 STEP 1
330  FOR R = 1 TO 100 STEP 1
```

2 Enter a conditional statement inside each loop, after the GOSUB statement.

```
220  IF N$ = "9999" THEN 245
350  IF N$ = "9999" THEN 375
```

3 Enter dummy data as shown:

```
9000  DATA  "9999", "99", "99", "99"
9005  DATA  "9", "9", "9", "9"
```

When the dummy data is READ, the program will branch to another statement.

By placing the DUMMY DATA statements at high statement numbers, you may insert additional DATA statements for additional students, if required.

SAVE as STUDENT DETAILS

LIST and RUN

PRINT LIST and RUN

Centring a heading

Instead of printing the heading at the left-hand margin, you may prefer to centre it over the table.

Assume that you want to print on an 80-space line.

You could:

1 count the characters and spaces in the heading (PERSONAL DETAILS = 16)

2 subtract the number from the line length (80 − 16 = 64)

3 halve that number (64/2 = 32)

4 print heading at TAB position 32.

However, the system will do all of this for you, by using the:

LEN (Length) statement — to determine the number of characters in the variable

INT (Integer) statement — to return a whole number.

Enter these statements:

```
170  GOSUB 650
290  GOSUB 650
650  REM  SUBROUTINE TO CENTRE HEADINGS
660 L =  LEN (H$)              (refer step 1 above)
670 T =  INT (80 - L) / 2      (refer steps 2 and 3 above)
680  PRINT  TAB( T);H$         (refer step 4 above)
690  RETURN
```

Delete these statements:

180 and 300

SAVE as STUDENT DETAILS

LIST and RUN

PRINT LIST and RUN

Using the LEFT$, MID$, RIGHT$ statements

With these statements you may select specified characters from a variable to assist your programming.

Example

X$ = EXAMPLE	
L$ = LEFT$ (X$,2)	Returns first two characters — EX
R$ = RIGHT$ (X$,3)	Returns last three characters — PLE
M$ = MID$ (X$,2,4)	Returns four characters, starting with the second character — XAMP

When you run your program, there may be occasions when you do not want to print both lists. The statements below will give you that choice.

Enter these statements:

```
130  PRINT "DO YOU WANT TO PRINT PERSONAL DETAILS?": PRINT
140  GOSUB 600
150  IF  LEFT$ (Y$,1) = "Y" THEN 160
155  GOTO 250
250  PRINT "DO YOU WANT TO PRINT STUDENT ENROLMENT?": PRINT
260  GOSUB 600
270  IF  LEFT$ (Y$,1) = "Y" THEN 280
275  GOTO 375
600  REM  SUBROUTINE FOR PRINTING CHOICE
610  PRINT "ENTER YES OR NO AND PRESS RETURN": PRINT
620  INPUT Y$
630  RETURN
```

SAVE as STUDENT DETAILS

LIST and RUN

PRINT LIST and RUN

This is a useful function — when the program is run, some operators would type "YES", but others might type only "Y". This example covers both types of input.

Using the VAL statement

The VAL statement will convert a string variable into a numeric value. If you want to print the actual age of students, you could do this by comparing each student's year of birth with the current year.

To convert sections of string variables for birthdate (B$) and current year (D$) to numeric values:

B = VAL(RIGHT$(B$,2)) Returns year of birth

D = VAL(RIGHT$(D$,2)) Returns current year

To find age of student:

A = D − B

Enter these statements:

```
380  PRINT "DO YOU WANT TO PRINT STUDENT AGES?": PRINT
390  GOSUB 600
400  IF  LEFT$ (Y$,1) = "Y" THEN 410
405  GOTO 9999
410  H$ = "STUDENT AGES"
420  GOSUB 650
425  PRINT : PRINT D$: PRINT
430  PRINT "NAME","BIRTHDATE","AGE","ADDRESS"
435  PRINT
440  RESTORE
450  FOR R = 1 TO 100 STEP 1
460  GOSUB 700
470  IF N$ = "9999" THEN 9999
480  B =  VAL ( RIGHT$ (B$,2))
490  D =  VAL ( RIGHT$ (D$,2))
500  A = D - B
510  PRINT N$,B$,A,A$
520  NEXT R
```

SAVE as STUDENT DETAILS

LIST and RUN

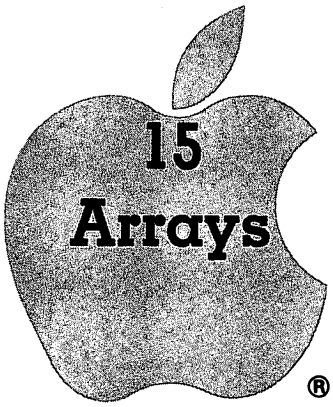
PRINT LIST and RUN

Notes:

1 You must keep this program on disk or cassette, as you will be working with it again in the next chapter.

2 You could also compare the current month with the birth month, to get a more accurate age, by using the MID$ statement.

3 Students' marks could be entered as data (instead of E or N) to obtain a printed list of actual results.



An array is simply a list that stores the values for a number of variables.

Using the data from the previous chapter, the array (N$) could be represented as a one-dimensional array.

One-dimensional array

Col 1	Col 2	Col 3	Col 4	Col 5	Col 6	Col 7	Col 8
Name	Address	Phone	Birthdate	Class	English	Maths	Science

The array N$ now has eight columns, called **elements**, and they are referred to by the number of the element, eg

Name N$(1)
Address N$(2)
Phone N$(3)
Birthdate N$(4)
Class N$(5)
English Can you identify
Maths these elements?
Science

If you want to use an array to read and store all the information for one student:

```
FOR E = 1 TO 8      (number of elements)
READ N$(E)          (to read element E)
NEXT E              (increase E by 1)
```

These statements form a loop — the first time through the loop E = 1 so the first data item (name) will be read and stored as N$(1). The second time through the loop, E = 2, so the second data item (address) will be read and stored as N$(2).

This loop will continue until all data items have been read and stored under their own variable names.

Two-dimensional array

If you want to read and store data for more than one student, you must use a two-dimensional array (sometimes called a **matrix**).

This diagram represents the two-dimensional array (N$).

	Col 1 Name	Col 2 Address	Col 3 Phone	Col 4 Birthdate	Col 5 Class	Col 6 English	Col 7 Maths	Col 8 Science
Row 1	1,1	1,2	1,3	1,4	1,5	1,6	1,7	1,8
Row 2	2,1	2,2	2,3	2,4	2,5	2,6	2,7	2,8
Row 3	3,1	3,2	3,3	3,4	3,5	3,6	3,7	3,8

All details for student 1 are stored in row 1.

All details for student 2 are stored in row 2.

All details for student 2 are stored in row 3.

Details	*Student 1*	*Student 2*	*Student 3*
Name	N$(1,1)	N$(2,1)	N$(3,1)
Address	N$(1,2)	N$(2,2)	N$(3,2)
Phone	N$(1,3)	N$(2,3)	N$(3,3)
Birthdate	N$(1,4)	N$(2,4)	N$(3,4)
Class	N$(1,5)		
English	N$(1,6)		
Maths	N$(1,7)	Can you complete	
Science	N$(1,8)	the missing variables?	

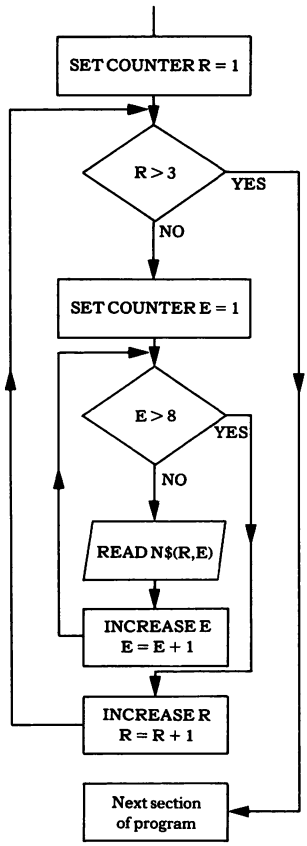
Nested loop

To use a two-dimensional array to read and store information for more than one student, you will need to use nested loops, ie one loop which works inside another loop, eg

```
FOR R = 1 TO 3          (number of rows)
FOR E = 1 TO 8          (number of elements)
```

READ N$(R,E)	(read row R, element E)
NEXT E	(increase E by 1)
NEXT R	(increase R by 1)

The nested loop is represented in a flowchart as:



The inner loop (E) which reads the elements, is nested inside the outer loop (R) which reads the rows.

The two loops must never cross each other, for such a condition would stop a program RUN.

Now let us adjust your program, Student Details, to use a two-dimensional array.

1 LOAD STUDENT DETAILS

2 Save under another name, eg
SAVE ARRAYS
This will allow you to keep one program, Student Details, and a separate program, Arrays.

3 Delete these statements:

40 to 100	variables description
210, 220, 230	portion of first loop
320	RESTORE statement
340, 350, 360	portion of second loop
440	RESTORE statement
460, 470, 480, 510	portion of third loop
700, 710, 720	subroutine to read DATA

4 If the two-dimensional array contains more than ten rows, then you must use a **DIMENSION** statement to set the size of the array, ie the maximum number of rows and the maximum number of elements.

For your program you do not need a **DIMENSION** statement, but you may insert one for practice.

Enter:

```
5  DIM N$(100,8)
```

5 Enter these statements to identify the variables:

```
35  REM  NAME        N$(R,1)
40  REM  ADDRESS     N$(R,2)
45  REM  PHONE       N$(R,3)
50  REM  BIRTHDATE   N$(R,4)
55  REM  CLASS       N$(R,5)
60  REM  ENGLISH     N$(R,6)
65  REM  MATHS       N$(R,7)
70  REM  SCIENCE     N$(R,8)
```

6 Enter these statements to read and store all data:

```
75  FOR R = 1 TO 100 STEP 1
80  FOR E = 1 TO 8 STEP 1
85  READ N$(R,E)
90  IF N$(R,1) = "9999" THEN 110
95  NEXT E
100  NEXT R
```

7 Enter these statements to **PRINT** details for 'Personal Details':

```
210  IF N$(R,1) = "9999" THEN 245
220  PRINT N$(R,1),N$(R,3),N$(R,4),N$(R,2)
```

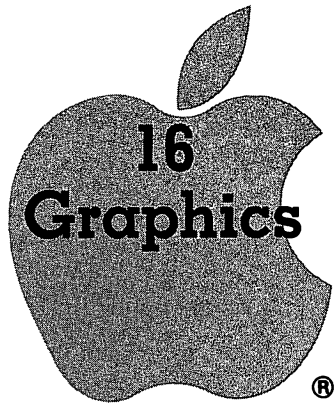
8 Enter these statements to **PRINT** details for 'Student Enrolment':

```
340  IF N$(R,1) = "9999" THEN 375
350  PRINT N$(R,1),N$(R,5),N$(R,6),N$(R,7),
     N$(R,8)
```

9 Enter these statements to **PRINT** details for 'Student Ages':

```
460  IF N$(R,1) = "9999" THEN 9999
480  B =  VAL ( RIGHT$ (N$(R,4),2))
510  PRINT N$(R,1),N$(R,4),A,N$(R,2)
```

SAVE as **ARRAYS**
LIST and **RUN**
PRINT **LIST** and **RUN**

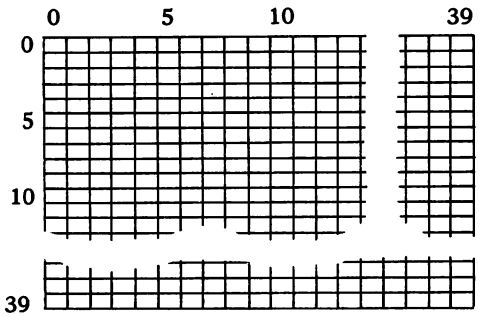


Graphics allow you to draw shapes on the screen with dots of light. This chapter refers to low-resolution graphics.

Changing to graphics

GR tells the computer that you will be using most of the screen to draw a shape. The screen will then go black except for four lines at the bottom which will allow you to see the text you are typing. This lower area is called a **text window.**

The top of the screen (the graphic screen) is divided into a grid with 40 columns across and 40 rows down.



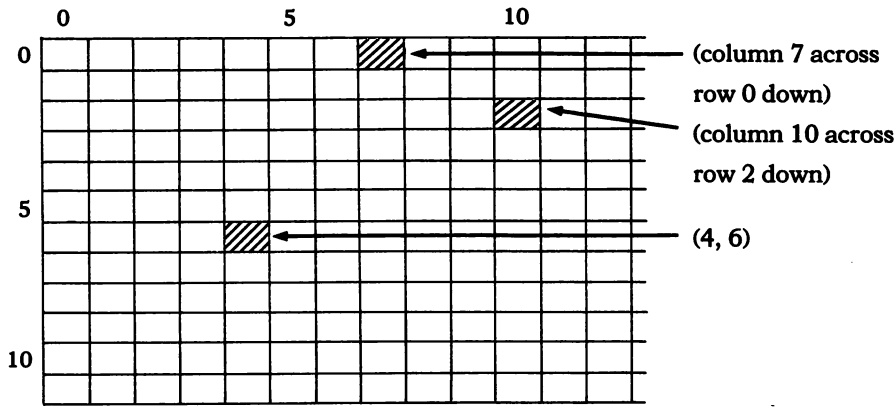
Each block is numbered starting from
0 to 39 across
0 to 39 down
Each point is actually a small rectangle.

If you plot a point outside these numbers you will get an error message: ILLEGAL QUANTITY ERROR.

Identifying the location of a point

When listing the location of a point, always list the columns **across** and then the rows **down.**

Only part of a graphic screen is listed below—this grid does not actually appear on the screen.



Colours available

The Apple//e has colour graphics. There is a choice of sixteen colours and each colour has its own number:

0 Black	6 Medium blue	12 Green
1 Magenta	7 Light blue	13 Yellow
2 Dark blue	8 Brown	14 Aqua
3 Purple	9 Orange	15 White
4 Dark green	10 Grey (different from 5)	
5 Grey	11 Pink	

If you do not specify a different colour the computer will automatically choose black, ie 0. Then, when you plot a point on the screen nothing will appear, as you have plotted a black point on a black background.

Choosing a colour

COLOR=12 (note the spelling of COLOR)
This will set the colour to green.

If you have a black and white television monitor or set you can still select a colour and the drawing will appear in shades of white to black (ie grey).

Immediate execution of commands

Setting up graphics

1. Turn machine on.
2. Prompt and cursor will appear.
3. Type GR and press RETURN. (Notice that cursor has moved to bottom part of the screen.)

4 Type COLOR=12 and press RETURN. (Colour selected is green.)

Plotting a point

5 Type PLOT 5, 9 and press RETURN. (Be sure to type the comma between the two numbers.)
This point will light up on the screen. (Column 5 across row 9 down.)
6 Plot another point—type PLOT 10,14 .
7 Change the colour—type COLOR=3 (ie purple).
8 Plot a point at column 15, row 20.
9 Plot 40,40—note error message—blocks are numbered 0 to 39 across and 0 to 39 down.

Cancelling a plotted point

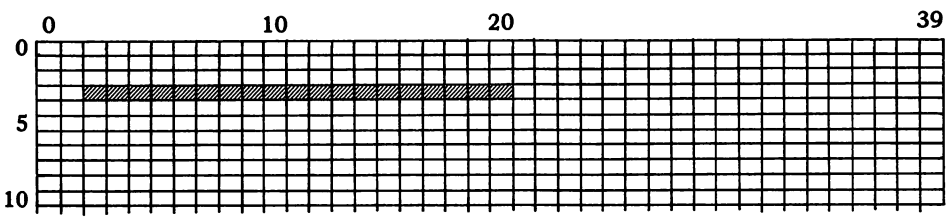
If a point is in the wrong position follow these steps:

1 Set colour to black (ie COLOR=0)
2 Re-enter **exactly** the incorrect plot. Try cancelling your last plot:
Check: COLOR=0
PLOT 15,20
3 Now reset the colour to the required number.

Drawing horizontal lines

To light up a line **across** the screen:
HLIN 2,20 AT 3
This statement tells the computer:
- to draw a line across the screen starting at column 2 and finishing at column 20
- the line should be on row 3 (ie 4 rows from the top of the screen).

Remember: This line is really a series of plotted points.



Enter the above statement.

Change the colour on the screen.

Enter two more horizontal lines. (**Remember:** 0 to 39 across, 0 to 39 down)

Cancel the first line, ie COLOR = 0
HLIN 2,20 AT 3

Remember: Reset colour after deletion.

Drawing vertical lines

To light up a line **down** the screen:
VLIN 3,6 AT 10
This statement tells the computer:
- to draw a line down the screen starting at row 3 and finishing at row 6
- the line should be at column 10 across the screen.

Remember: This line is really a series of plotted points.
Enter the above statement.

Change the colour on the screen.

Enter two more vertical lines.

Cancel the first vertical line, ie COLOR=0
VLIN 3,6 AT 10

Remember: Reset the colour after deletion.

Returning to full screen text

When using graphics you have only the text window for display of instructions. If you wish to return to text on the full screen:

Type TEXT and press RETURN—this changes the display from graphics to characters.

Because text and graphics both use the same area of memory you will now see a screen full of 'at' (@) signs.

To clear the screen, type HOME .

Summary

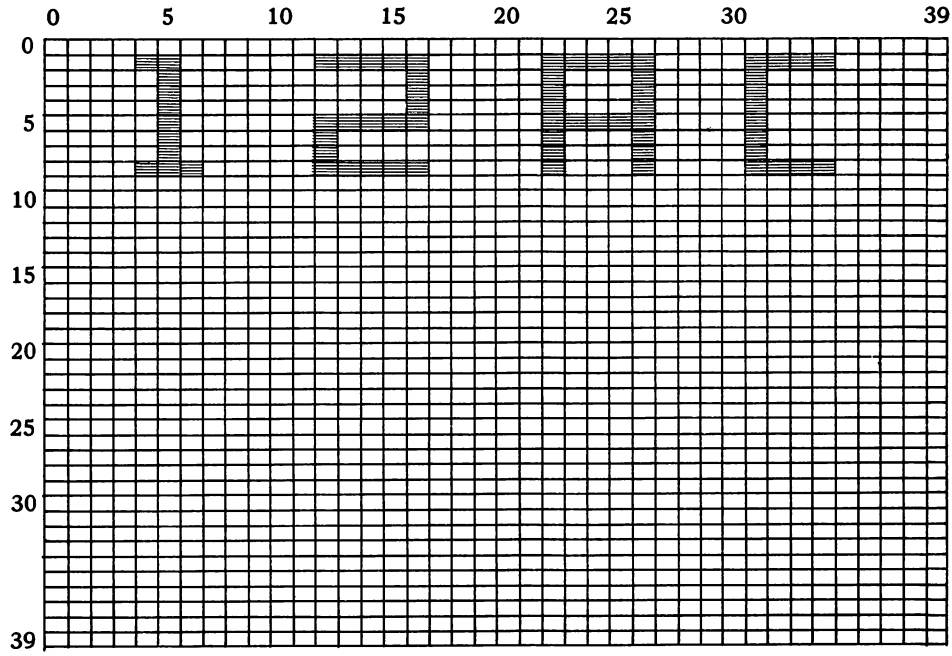
1 GR
- clears screen for graphics
- sets up text window—ie 4 lines at bottom of screen
- sets colour at 0—ie black.

2 COLOR=10—sets colour from 0 to 15.

3 PLOT 5, 10—plots one point.
HLIN 5,10 AT 6—plots horizontal line.
VLIN 3,20 AT 10—plots vertical line.

4 TEXT—changes from graphics to text.

5 HOME—clears screen. Try to plot the following numbers and letters:



In the space at the bottom of the above grid draw in your name and then plot your name on the screen.

Deferred execution of commands

By inserting a number beside each statement the computer can store the commands so that you can LIST and RUN the program and then, if desired, you can save it on to disk. Now you have created a BASIC program to draw the shape.

Input the following program *exactly* as shown:

```
NEW
10  GR
20  COLOR= 13
30  HLIN 2,6 AT 21
40  HLIN 10,14 AT 21
50  HLIN 18,22 AT 21
60  HLIN 33,36 AT 21
70  HLIN 3,5 AT 25
80  HLIN 11,13 AT 25
90  HLIN 19,21 AT 25
100  HLIN 34,36 AT 25
110  HLIN 27,29 AT 28
120  HLIN 34,36 AT 28
130  VLIN 22,28 AT 2
140  VLIN 22,28 AT 6
```

(continued)

```
150  VLIN 22,28 AT 10
160  VLIN 22,25 AT 14
170  VLIN 22,28 AT 18
180  VLIN 22,25 AT 22
190  VLIN 21,28 AT 26
200  VLIN 22,28 AT 33
9999  END
```

LIST

RUN

If you wish to LIST the program again, type TEXT then LIST

Making graphics flash

Add the following statement to your previous program:

GO TO 10

Now LIST and then RUN the program again.

Remember: To stop continuous loop—CTRL–C
To continue program—type CONT

The word appears to be blinking—do you know why?

We have instructed the computer to go back to statement 10 which was GR. This statement automatically makes the colour equal to 0 (ie black) which means that your words have disappeared.

Suggestions

1. Use graph paper to draft your picture.
2. Write your program from the graph paper.
3. Enter the program. LIST, RUN.
4. To LIST the program again, type TEXT for full screen display.

Mystery programs

Enter the following programs:

Mystery no 1

NEW

```
10  GR
20  COLOR= 2
30  HLIN 17,29 AT 36
40  HLIN 16,30 AT 35
50  HLIN 15,31 AT 34
60  HLIN 14,31 AT 33
70  HLIN 9,13 AT 34
80  VLIN 29,33 AT 8
90  HLIN 9,10 AT 28
100  PLOT 11,29
110  PLOT 12,30
120  PLOT 13,31
130  PLOT 14,32
140  COLOR= 15
150  VLIN 18,32 AT 25
160  VLIN 18,32 AT 24
170  VLIN 18,32 AT 23
180  VLIN 18,32 AT 22
190  VLIN 18,32 AT 21
200  HLIN 21,25 AT 17
```

(continued)

```
210  HLIN 21,24 AT 16
220  HLIN 21,23 AT 15
230  HLIN 21,22 AT 14
240  PLOT 21,13
250  COLOR= 9
260  VLIN 11,12 AT 23
270  HLIN 23,24 AT 10
280  HLIN 23,23 AT 9
290  HLIN 23,24 AT 8
300  HLIN 22,24 AT 7
310  HLIN 22,24 AT 6
320  HLIN 22,23 AT 5
330  VLIN 2,4 AT 23
9999  END
```

Mystery no 2

NEW

```
5  GR
8  COLOR= 12
10  PLOT 24,2
20  PLOT 23,2
30  HLIN 22,23 AT 4
40  HLIN 21,22 AT 5
50  HLIN 20,21 AT 6
60  HLIN 20,21 AT 7
70  VLIN 8,11 AT 19
80  VLIN 8,11 AT 20
90  HLIN 12,15 AT 10
100  HLIN 24,27 AT 10
110  HLIN 11,16 AT 11
120  HLIN 23,28 AT 11
125  COLOR= 13
130  HLIN 10,29 AT 12
140  HLIN 10,29 AT 13
150  HLIN 9,30 AT 14
160  HLIN 9,30 AT 15
165  COLOR= 9
170  HLIN 9,30 AT 16
180  HLIN 8,30 AT 17
190  HLIN 8,29 AT 18
200  HLIN 8,29 AT 19
210  HLIN 8,28 AT 20
220  COLOR= 11
230  FOR I = 21 TO 25
240  HLIN 8,28 AT I
250  NEXT I
260  COLOR= 3
270  HLIN 8,29 AT 26
280  HLIN 8,29 AT 27
290  HLIN 8,30 AT 28
300  HLIN 8,30 AT 29
310  HLIN 9,29 AT 30
320  HLIN 9,29 AT 31
330  COLOR= 6
340  HLIN 10,29 AT 32
350  HLIN 10,28 AT 33
360  HLIN 11,27 AT 34
370  HLIN 12,27 AT 35
380  HLIN 14,25 AT 36
390  HLIN 16,18 AT 37
400  HLIN 21,23 AT 37
9999  END
```

Mystery no 3

Alter mystery no 2 as shown:

```
190  HLIN 8,31 AT 18
200  HLIN 8,31 AT 19
210  HLIN 8,31 AT 20
220  HLIN 8,31 AT 21
230  HLIN 8,31 AT 22
240  HLIN 8,31 AT 23
250  HLIN 8,31 AT 24
260  HLIN 8,31 AT 25
270  HLIN 8,31 AT 26
280  HLIN 8,31 AT 27
400  FOR L = 1 TO 900: NEXT L
410  COLOR= 0
420  VLIN 17,28 AT 31
430  FOR L = 1 TO 900: NEXT L
440  VLIN 18,27 AT 30
450  FOR L = 1 TO 900: NEXT L
460  VLIN 19,26 AT 29
470  FOR L = 1 TO 900: NEXT L
480  VLIN 21,24 AT 28
```

Mystery No 4

NEW

```
10  GR
20  COLOR= 12
30  PLOT 19,1
40  HLIN 18,20 AT 2
50  HLIN 18,20 AT 3
60  HLIN 17,21 AT 4
70  HLIN 17,21 AT 5
80  HLIN 16,22 AT 6
90  HLIN 16,22 AT 7
100  HLIN 15,23 AT 8
110  HLIN 15,23 AT 9
120  HLIN 14,24 AT 10
130  HLIN 14,24 AT 11
140  HLIN 13,25 AT 12
150  HLIN 13,25 AT 13
160  HLIN 12,26 AT 14
170  HLIN 12,26 AT 15
180  HLIN 11,27 AT 16
190  HLIN 11,27 AT 17
200  HLIN 10,28 AT 18
210  HLIN 10,28 AT 19
220  HLIN 9,29 AT 20
230  HLIN 9,29 AT 21
240  HLIN 8,30 AT 22
250  HLIN 8,30 AT 23
260  HLIN 7,31 AT 24
270  HLIN 7,31 AT 25
280  HLIN 6,32 AT 26
290  HLIN 6,32 AT 27
300  HLIN 5,33 AT 28
310  HLIN 5,33 AT 29
320  COLOR= 8
330  VLIN 30,34 AT 18
340  VLIN 30,34 AT 19
350  VLIN 30,34 AT 20
360  COLOR= 6
370  FOR X = 35 TO 39 STEP 1
380  HLIN 16,22 AT X
390  NEXT X
400  FOR X = 1 TO 200
410  COLOR= 15
420  PLOT 19,8
430  PLOT 23,11
440  PLOT 22,22
450  PLOT 9,27
460  COLOR= 13
470  PLOT 12,21
480  PLOT 15,24
490  PLOT 28,25
500  COLOR= 9
510  PLOT 19,3
520  PLOT 22,17
530  PLOT 15,15
540  COLOR= 11
550  PLOT 25,28
560  PLOT 27,19
570  PLOT 18,12
580  COLOR= 7
590  PLOT 19,26
600  PLOT 30,28
610  PLOT 13,28
620  COLOR= 0
630  PLOT 19,8
640  PLOT 23,11
650  PLOT 22,22
660  PLOT 9,27
670  PLOT 12,21
680  PLOT 15,24
690  PLOT 28,25
700  PLOT 19,3
710  PLOT 22,17
720  PLOT 15,15
730  PLOT 25,28
740  PLOT 27,19
750  PLOT 18,12
760  PLOT 19,26
770  PLOT 30,28
780  PLOT 13,28
790  NEXT X
9999  END
```

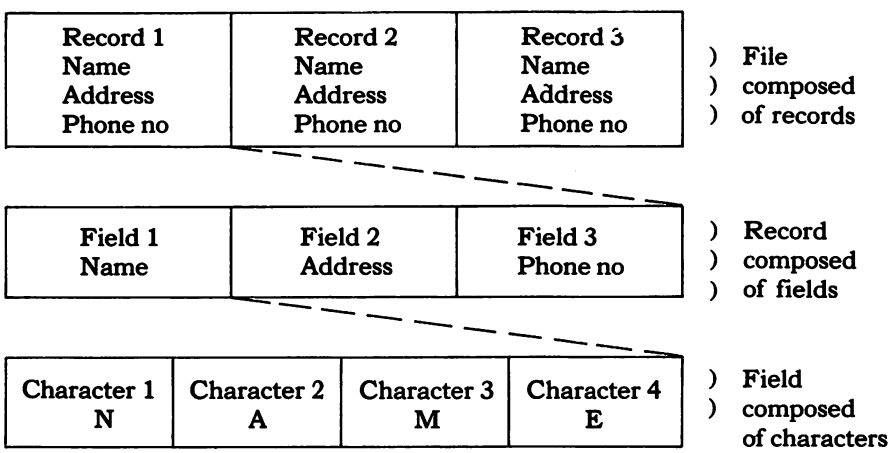
What is a textfile?

A file is a collection of **records** which is stored on some medium in a specific order. Each record may contain one or more pieces of **data.** Each piece of data may be referred to as a **field.**

A telephone directory is a **file** in which the name, address and telephone number of each subscriber is a separate record. The name, address and telephone number represent different fields. In this case the recording medium is paper.

As files contain a number of data items, this collection of data may also be thought of as a very simple form of a database. The database may be used to provide information of various types. It is also possible to **update** the file (ie change, insert or delete data in a record) without affecting any other records.

Using the telephone directory as an example, that database could be used to provide a list of all surnames starting with WI . . . , a separate list of all telephone numbers starting with 268 . . . , or to update any record.



Files can also be created, stored and retrieved using the Apple//e and a magnetic recording medium.

Already you have been creating, storing and retrieving files using NEW, SAVE and LOAD with respect to the programs you have written.

Let's now look at the commands required to store data, which is not in a program format, on to a magnetic recording medium, ie disk or cassette tape. Data stored in this manner is usually referred to as a **textfile.**

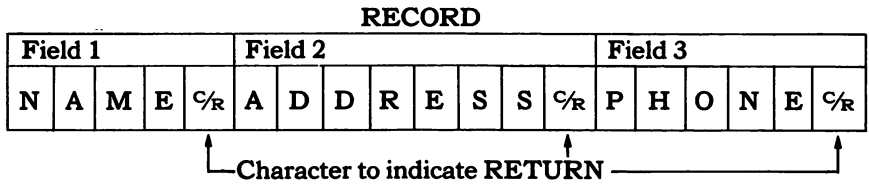
Two types of textfiles

Sequential files

Sequential textfiles may be used with both disks and cassette tapes. They are textfiles in which records, usually of differing lengths, are stored in a serial manner, ie one after the other.

One disadvantage of sequential files is that to obtain the data from, say, the tenth record, the previous nine records must be read first. The sequential file can only be read in one direction, ie from the first record through to the last record. Thus, it is not possible to read the tenth record and then go back and read the first record.

Sequential files are ideal for use where the majority of the records have to be read or updated. They are also space saving, in that fields usually do not contain blank spaces.



Random-access files

Records within a random-access file must be of equal length. Each record is given a specific record number. Thus it is possible to read or update records in a random manner, ie the data in, say, record 10 may be read, then record 2, then record 4, then record 1.

From this it should be obvious that random-access files can only be used with disks, and not with cassette tapes.

Because records must be of equal length, there may be unused space in each record on the disk.

Example:

Field 1	Name	Allow for 15 characters
Field 2	Address	Allow for 25 characters
Field 3	Phone	Allow for 10 characters
Record length		50 characters

However, not all names will be 15 characters in length, nor will all addresses be 25 characters in length, so some space in each record will not be used.

RECORD

Char. 1 Char. 50

RICHARDSON..... 7 Mill St Wynnum.....268.1234
WILEY ..... 13 Baker St Mornington 399.2468
McDOUGALL.....284 More St Springwood.....341.5555

Method of using textfiles

It is important to realise that, when working with a textfile, you will actually be using *two* files. One file is a program file, which is used to create and access a separate textfile.

The program file will instruct the system to OPEN a textfile; WRITE data to that textfile, READ data from that textfile, or update data in that textfile; and finally to CLOSE the textfile.

Compare this with the following example of accessing data contained in a filing cabinet.

Filing cabinet	*Textfile*
Open drawer	OPEN file
Place data in file	WRITE data to file
Obtain data from file	READ data from file
Change data in file	Update data in file
Close drawer	CLOSE file

Remember that you can't obtain any data until you have opened the drawer/file and, for safety reasons, you must close the drawer/file when you have finished accessing the data.

Sequential files

Creating a sequential file: OPEN WRITE PRINT CLOSE

First let's create a sequential file (CLASS-LIST) and write data to that file.

To enter statement 20, type D$ = " (then depress and hold CONTROL and type letter D then release both keys) and then type the second quotation mark. As the CTRL–D command does not display on the screen, the complete command will appear as D$ = "". The REMark following this command will remind you of what does not appear on the screen.

Enter:
NEW

```
10  REM  PROGRAM NAME CLASS-LIST CREATE
20 D$ = "": REM  CTRL-D
30  PRINT D$;"OPEN CLASS-LIST"
40  PRINT D$;"WRITE CLASS-LIST"
50  PRINT "ALBERT"
60  PRINT "BENJAMIN"
70  PRINT "CAROL"
80  PRINT "LAURIE"
90  PRINT "MONICA"
100  PRINT "SUE"
110  PRINT "RUTH"
120  PRINT D$;"CLOSE CLASS-LIST"
9999  END
```

LIST RUN SAVE as CLASS-LIST CREATE 1
Now check the catalog — there should be two new files, one named CLASS-LIST CREATE which is prefixed by the letter A to indicate Applesoft BASIC, and one named CLASS-LIST prefixed by the letter T to indicate textfile.

Line 20 established the DOS (Disk Operating System) command of CTRL–D which enables the program to pass instructions to the DOS in lines 30, 40 and 120.

Line 30 initiates a number of actions:

- Opens a file called CLASS-LIST.
- Lists the file in the catalog.
- Reserves memory space (called a file buffer) to handle input and output of the textfile.
- Sets up slot, drive and/or volume parameters. (Instructions on these parameters are not included in this chapter — they are mentioned here only to make you aware of their existence. When you become more familiar with textfiles you should consult the DOS Reference Manual for a complete explanation of parameters.)

Line 40 writes the output from the following PRINT statements (50 – 110) to the textfile CLASS-LIST *instead* of to the screen. Thus, when you run the program, the data within quotation marks will *not* be displayed on the screen.

Line 120 closes the textfile and also clears the memory space previously allocated as a file buffer.

The textfile (CLASS-LIST) consists of items of data separated by a RETURN character (which you entered at the end of each statement). Each data item (in this case a name) and its following RETURN character, is called a FIELD. The FIELDS in this case are of different lengths. The system can recognise the end of each field by the RETURN character.

Retrieving data: OPEN READ INPUT CLOSE

Load and list CLASS-LIST CREATE.

Add these statements to your program.

```
200  REM  RETRIEVE CLASS-LIST
210  PRINT D$;"OPEN CLASS-LIST"
220  PRINT D$;"READ CLASS-LIST"
230  FOR I = 1 TO 7 STEP 1
240  INPUT A$(I)
250  NEXT I
260  PRINT D$;"CLOSE CLASS-LIST"
```

LIST RUN SAVE as CLASS-LIST CREATE 2

Line 210 opens the textfile CLASS-LIST.

Line 220 instructs DOS that the followong INPUT statements refer to the textfile and not to the keyboard. Each INPUT will cause one complete field to be transferred from the textfile through the file buffer to the memory. Thus the lines 230 to 250 will store the names in an array (A$) whose elements are A$(1), A$(2), etc.

Don't be dismayed that the names do not appear on the screen when you RUN this program. They are stored in memory! If you wish to see the names displayed on the screen, you could insert this line:

245 PRINT A$(1)

Adding data: OPEN APPEND PRINT CLOSE

Enter this program:

NEW

```
10  REM  PROGRAM NAME CLASS-LIST ADD
20 D$ = "": REM  CTRL-D
30  PRINT D$;"APPEND CLASS-LIST"
40  PRINT D$;"WRITE CLASS-LIST"
50  PRINT "MARY"
60  PRINT "ALEXANDER"
70  PRINT "DIANA"
80  PRINT "WALTER"
90  PRINT D$;"CLOSE CLASS-LIST"
9999  END
```

LIST — check program very carefully. Each time this program is run, it will add the data to the file.

RUN and SAVE as CLASS-LIST ADD

Line 30 uses the command APPEND instead of OPEN. If you need to add further dates to a file, you must first find the end of the existing file. This may be found by:

- using the OPEN command, which means that DOS will start at the beginning of the file and then read each record through to the last record. This may be laborious and time-consuming.

- using the APPEND command, which means that DOS will locate the first *unused* character position. Remember that in a sequential file there are no unused spaces, so the first *unused* character position can only be at the end of the file.

Retrieving data: OPEN READ INPUT CLOSE

Enter:

NEW

```
10  REM  PROGRAM NAME CLASS-LIST READ
20  D$ = "": REM  CTRL-D
30  DIM A$(15)
40  PRINT D$;"OPEN CLASS-LIST"
50  PRINT D$;"READ CLASS-LIST"
60  FOR I = 1 TO 11 STEP 1
70  INPUT A$(I)
80  PRINT A$(I)
90  NEXT I
100  PRINT D$;"CLOSE CLASS-LIST"
9999  END
```

Notice statement 30, which DIMensioned the array (A$), enabling it to hold up to fifteen (15) elements.

Remember: INPUT transfers data to memory; PRINT allows screen display.

LIST RUN SAVE as CLASS-LIST READ

Retrieving selected data: POSITION READ INPUT CLOSE

If for some reason you want to access only part of the file, this is possible by using the POSITION command. Remember that OPEN automatically gives access to the beginning of the file. POSITION allows access to any selected field within a textfile.

Enter:

NEW

```
10  REM  PROGRAM NAME CLASS-LIST POSITION
20  D$ = "": REM  CTRL-D
30  PRINT D$;"OPEN CLASS-LIST"
40  PRINT D$;"POSITION CLASS-LIST,R4"
50  PRINT D$;"READ CLASS-LIST"
60  INPUT A$
70  PRINT A$
80  PRINT D$;"CLOSE CLASS-LIST"
9999  END
```

LIST RUN SAVE as CLASS-LIST POSITION

Did you notice that, by using R4 as the selected field, the system printed the fifth name (MONICA)? This happened because the first record is actually labelled as zero (0). You could also have accessed the file starting at any selected field and then inserting a FOR and NEXT loop and an array if you wished to view a number of records, as in previous examples.

Deleting a textfile: DELETE

If you wish to delete an entire textfile, simply type CATALOG to check the filename, then DELETE filename .

However, there may be occasions on which you want to use the same filename, but to enter all new data. Unless you delete the previous contents, the new data will be overwritten on the old data. As the fields in a sequential file are of different lengths, the new data may be less than the old data. In this case, the new data would overwrite the old data from the beginning of the file, but some of the old data could be left at the end of the file. To overcome this problem you should delete the file before you store new data.

The following statements will achieve this:

```
NEW
10  REM  PROGRAM NAME CLASS-LIST DELETE
20  D$ = "": REM  CTRL-D
30  PRINT D$;"OPEN CLASS-LIST"
40  PRINT D$;"DELETE CLASS-LIST"
50  PRINT D$;"OPEN CLASS-LIST"
60  PRINT D$;"CLOSE CLASS-LIST"
9999  END
```

LIST RUN SAVE as CLASS-LIST DELETE

Line 30 opened the file, line 40 deleted the file, line 50 opened a new file with the same filename and line 60 closed the file. Thus you now have an empty file, ie no data is stored in the file.

Commas and semicolons

In files other than textfiles, the comma and semicolon may be used to print output in a specific format, ie

- comma causes column print
- semicolon causes close print.

However, this situation changes when data is stored in a textfile, then retrieved and displayed.

To illustrate these differences, enter the following program:

```
NEW
10  REM  PROGRAM NAME PUNCTUATION SIGNS
20 D$ = "": REM  CTRL-D
30  PRINT "PROGRAMME OUTPUT": PRINT
40  PRINT "JANUARY ";"FEBRUARY ";"MARCH"
50  PRINT "APRIL","MAY","JUNE"
60  PRINT 12;34;56
70  PRINT 78,90,24: PRINT
80  PRINT D$;"OPEN COMS AND SEMIS"
90  PRINT D$;"WRITE COMS AND SEMIS"
100  PRINT "JANUARY ";"FEBRUARY ";"MARCH"
110  PRINT "APRIL","MAY","JUNE"
120  PRINT 12;34;56
130  PRINT 78,90,24
140  PRINT D$;"CLOSE COMS AND SEMIS"
150  PRINT : PRINT
200  PRINT "TEXT FILE OUTPUT": PRINT
210  PRINT D$;"OPEN COMS AND SEMIS"
220  PRINT D$;"READ COMS AND SEMIS"
230  FOR I = 1 TO 2 STEP 1
240  INPUT M$(I)
250  PRINT M$(I)
260  NEXT I
270  FOR C = 1 TO 4 STEP 1
280  INPUT N(C)
290  PRINT N(C)
300  NEXT C
310  PRINT D$;"CLOSE COMS AND SEMIS"
9999  END
```

LIST RUN SAVE as TEXTFILE PUNCTUATION 1

Compare the two outputs (program and textfile) and notice the differences in format. Not what you expected!

You will remember that earlier in this chapter it was mentioned that fields in a textfile are separated by the RETURN character. When you entered lines 100 and 110 only two fields were actually entered. This explains why the loop counter in line 230 was set to end at 2.

The comma and semicolon used with numeric data, as in lines 120 and 130, also affect the number of fields entered, hence the loop counter (4) in line 270.

Punctuation signs can only be used when they are totally enclosed within quotations marks, and thus they should not be used to enter data to textfiles.

To overcome this problem, every data entry must be followed by a RETURN.

This can be accomplished by:

1 Using a separate PRINT for every data item, eg

PRINT "JANUARY" RETURN
PRINT "FEBRUARY" RETURN
PRINT 12 RETURN
PRINT 34 RETURN
etc

2 Using multiple PRINT, eg

```
100  PRINT "JANUARY": PRINT "FEBRUARY": PRINT "MARCH"
110  PRINT "APRIL": PRINT "MAY": PRINT "JUNE"
120  PRINT 12: PRINT 34: PRINT 56
130  PRINT 78: PRINT 90: PRINT 24
```

3 Other methods are also available, but for simplicity only the above methods should be used at this stage.

Now adjust your program (TEXTFILE PUNCTUATION 1) by inserting the lines 100 to 130 from section 2 above. You will also need to alter both loop counters to read 1 TO 6.

LIST RUN SAVE as TEXTFILE PUNCTUATION 2

Formatting output

If you want to print the output in a selected format, you must first READ the file, INPUT the data to memory, CLOSE the file, and then establish the PRINT format.

Enter

NEW

```
10  REM  PROGRAM NAME TEXTFILE FORMAT
20 D$ = "": REM  CTRL-D
30  PRINT D$;"OPEN COMS AND SEMIS"
40  PRINT D$;"READ COMS AND SEMIS"
50  FOR I = 1 TO 6 STEP 1
60  INPUT M$(I)
70  NEXT I
80  FOR C = 1 TO 6 STEP 1
90  INPUT N(C)
100  NEXT C
110  PRINT D$;"CLOSE COMS AND SEMIS"
120  PRINT : PRINT "PRINTING FORMAT": PRINT
130  PRINT M$(1);M$(2);M$(3)
140  PRINT M$(4),M$(5),M$(6)
150  PRINT N(1);N(2);N(3)
160  PRINT N(4),N(5),N(6)
9999  END
```

LIST RUN SAVE as TEXTFILE FORMAT

Using a subroutine to print or input data

Often it is more convenient to use a subroutine to print to or input from a textfile. Enter this program to print data to memory. Enter statement numbers exactly as shown.

```
NEW
10  REM  PROGRAM NAME TEXTFILE SUBROUTINE
20  REM  N$=NAME,S$=SUBJECT,M=MARK
30  D$ = "": REM  CTRL-D
40  PRINT D$;"OPEN CLASS-LIST"
50  PRINT D$;"DELETE CLASS-LIST"
60  PRINT D$;"OPEN CLASS-LIST"
70  FOR K = 1 TO 4 STEP 1
80  PRINT D$;"WRITE CLASS-LIST"
90  GOSUB 500
100  NEXT K
110  PRINT D$;"CLOSE CLASS-LIST"
120  GOTO 9999
500  PRINT "SHIRLEY": PRINT "ENGLISH": PRINT 86
510  PRINT "MARK": PRINT "MATHS": PRINT 51
520  PRINT "COLIN": PRINT "ENGLISH": PRINT 67
530  PRINT "MELISSA": PRINT "SCIENCE": PRINT 65
600  RETURN
9999  END
```

LIST RUN SAVE as TEXTFILE SUBROUTINE 1

Now add these statements to input the data to memory:

```
200  PRINT D$;"OPEN CLASS-LIST"
210  PRINT D$;"READ CLASS-LIST"
220  FOR K = 1 TO 4 STEP 1
230  GOSUB 700
240  NEXT K
250  PRINT D$;"CLOSE CLASS-LIST"
260  GOTO 9999
700  INPUT N$(K): INPUT S$(K): INPUT M(K)
710  RETURN
```

Delete line 120

LIST RUN SAVE as TEXTFILE SUBROUTINE 2

Now add these statements to format output from the data stored in memory as arrays N$(K), S$(K) and M(K).

```
300  PRINT "CLASS LIST": PRINT
310  PRINT "NAME","SUBJECT","MARK": PRINT
320  FOR K = 1 TO 4 STEP 1
330  PRINT N$(K),S$(K),M(K)
340  NEXT K
350  PRINT : PRINT
```

(continued)

```
360  PRINT "ENGLISH RESULTS": PRINT
370  PRINT "NAME","MARK": PRINT
380  FOR K = 1 TO 4 STEP 1
390  IF S$(K) <  > "ENGLISH" THEN 410
400  PRINT N$(K),M(K)
410  NEXT K
420  GOTO 9999
```

Delete statement 260

LIST RUN SAVE as TEXTFILE SUBROUTINE 3

In this program you have used the textfile in a similar manner to a database. Access to the textfile has allowed you to print two different lists.

Using keyboard to input and retrieve text data

In chapter 10 you used the INPUT statement to enter data into a program. That statement can also be used with textfiles.

Enter these statements, which provide a program that you can use to create a textfile, enter data into that textfile and then retrieve and print that data on the screen to verify that it has been recorded on the magnetic medium.

NEW

```
10   REM  PROGRAM NAME TEXTFILE INPUT
20   REM  T$=TEXTFILE NAME, I$=DATA INPUT, C= TOTAL NUMBER OF RECORDS
25   DIM I$(30)
30   D$ = "": REM  CTRL-D
40   C = 0
50   INPUT "ENTER TEXTFILE NAME ";T$
60   PRINT D$;"OPEN ";T$
70   PRINT D$;"DELETE ";T$
80   PRINT D$;"OPEN ";T$
90   INPUT "ENTER DATA OR END ";I$
100   IF I$ = "END" THEN 170
110   PRINT D$;"WRITE ";T$
120   PRINT I$
130   PRINT D$
140   C = C + 1
150   GOTO 90
170   PRINT D$;"CLOSE ";T$
200   REM  TO RETRIEVE DATA
210   PRINT D$;"OPEN ";T$
220   PRINT D$;"READ ";T$
230   FOR K = 1 TO C STEP 1
240   INPUT I$(K)
250   NEXT K
260   PRINT D$;"CLOSE ";T$
300   REM  TO PRINT DATA
310   PRINT : PRINT
320   PRINT T$: PRINT
330   FOR K = 1 TO C STEP 1
340   PRINT I$(K)
350   NEXT K
9999   END
```

Notes:

Line 25 dimensions the variable A$ to contain up to 30 data items.

Line 40 sets the local number of records (C) to zero.

Line 140 increases C by 1 each time a record is entered.

Line 90 allows keyboard entry of data.

Line 130 deflects output back to the screen.

Line 340 formats the output and will print each record on a separate line.

LIST and check this program thoroughly.

Save as TEXTFILE INPUT 1

1 RUN TEXTFILE INPUT 1. For the textfile name enter CRICKET TEAM . For the data enter eleven surnames, then enter END to signify end of data.

2 LIST TEXTFILE INPUT 1

Enter:

```
325   PRINT "MAKE","PRICE",: PRINT
330   FOR K = 1 TO C STEP 2
340   PRINT I$(K),I$(K + 1)
```

These changes will allow the output to be formatted in two columns, with column headings.

Save as TEXTFILE INPUT 2

RUN

For the textfile name, enter PRICE LIST .

For the data, enter the name (then RETURN) and price (then RETURN) for seven cars. Enter END to signify end of data.

3 LIST TEXTFILE INPUT 2

Change:

```
325   PRINT "NAME","NUMBER","WAGE": PRINT
330   FOR K = 1 TO C STEP 3
340   PRINT I$(K),I$(K + 1),I$(K + 2)
```

These changes will allow the output to be formatted in three columns, with column headings.

Save as TEXTFILE INPUT 3

RUN

For the textfile name, enter EMPLOYEE DETAILS .

Select and enter appropriate data, in the order: name, employee number, and wage. Use END to signify end of data.

4 Now! Let's see if you can:

a create a textfile (INSURANCE)

b store in that file details of name, premium and month due for six people

c retrieve the data and print a list headed INSURANCE, with columns for Name, Month Due and Premium.

Random-access files

All records within a random-access file are of equal length. Therefore care must be taken to ensure that the data entered does not exceed that length. If you want to enter, say:

Field 1 Name maximum of 15 characters
Field 2 Phone maximum of 10 characters

then the Record Length would be set at 25 characters plus one RETURN character for each field, making the Record Length 27 characters.

Each record is identified by a number, indicating its position in the text-file. The first record is zero, then 1, 2, etc.

The following program is presented in segments to help you understand how the program works. As you enter each segment, you should LIST and check carefully but do not RUN until the entire program has been entered.

NEW

```
10  REM  PROGRAM NAME RAN-ACC CREATE
20  REM  N$=NAME,P$=PHONE NUMBER,M=TOTAL
30  D$ = "": REM  CTRL-D
40  M = 0
50  PRINT D$;"OPEN RANDOM,L27"
60  PRINT D$;"DELETE RANDOM"
70  PRINT D$;"OPEN RANDOM,L27"
80  PRINT D$;"WRITE RANDOM,R0"
90  PRINT M
100  PRINT D$;"CLOSE RANDOM"
```

Statements 80 to 100 store in Record 0 the total number of records (M). At this stage there are no records, so M=0.

```
110  PRINT D$;"OPEN RANDOM,L27"
120  INPUT "ENTER NAME OR END ";N$
130  IF N$ = "END" THEN 200
140  INPUT "ENTER PHONE NUMBER ";P$
150  M = M + 1
160  PRINT D$;"WRITE RANDOM,R";M
170  PRINT N$: PRINT P$
180  PRINT D$
190  PRINT : GOTO 120
```

(continued)

```
200  PRINT D$;"WRITE RANDOM,R0"
210  PRINT M
220  PRINT D$;"CLOSE RANDOM"
```

Statements 110 to 220 allow data entry (name N$ and phone number P$) to numbered records and store in Record 0 the total number of records (M).

```
300  REM  TO RETRIEVE AND PRINT ALL DATA
310  PRINT D$;"OPEN RANDOM,L27"
320  PRINT D$;"READ RANDOM,R0"
330  INPUT M
340  FOR C = 1 TO M STEP 1
350  PRINT D$;"READ RANDOM,R";C
360  INPUT N$(C): INPUT P$(C)
370  NEXT C
380  PRINT D$;"CLOSE RANDOM"
```

Statement 330 supplies the total number of records entered (M).
Statement 340 sets a loop from 1 to the value of M.
Statement 350 reads the records from 1 to value of M.
Statement 360 stores the data in memory in arrays N$ and P$.
Remember that if there are more than ten elements in an array, it must be DIMensioned.

```
400  REM  TO PRINT ALL DATA
410  PRINT : PRINT "TELEPHONE LIST": PRINT
420  PRINT "RECORD","NAME","NUMBER": PRINT
430  FOR C = 1 TO M STEP 1
440  PRINT C,N$(C),P$(C)
450  NEXT C
460  PRINT
```

Statements 400 to 460 print a list from the arrays stored in memory, together with the record number.

```
500  REM  TO PRINT SPECIFIC RECORD
510  PRINT "TOTAL NUMBER OF RECORDS IS ";M
520  INPUT "ENTER RECORD NUMBER ";N
530  PRINT D$;"OPEN RANDOM, L27"
540  PRINT D$;"READ RANDOM, R";N
550  INPUT N$: INPUT P$
560  PRINT N,N$,P$
570  PRINT D$;"CLOSE RANDOM"
9999  END
```

Statement 520 allows you to input any number between 1 and M, then to access and print that selected record — in this example to the screen.
LIST and SAVE as RAN-ACC CREATE
RUN, inserting relevant data (selected by you), and END to signify end of data.

Now that you have stored data in the textfile, you may wish to retain that textfile and perhaps add additional data. However, if you RUN the program RAN-ACC CREATE again, the value of M would be reset to zero by statement 40, and the contents of the textfile would be deleted by statement 60.

To overcome this problem, insert the following statements which will allow a choice of deleting the textfile, adding data, printing output or exiting the program. This choice is called a **menu**.

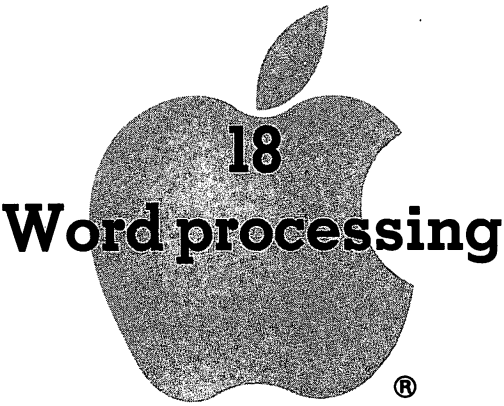
```
31  PRINT "DO YOU WANT TO: ": PRINT
32  PRINT "1 DELETE TEXTFILE": PRINT : PRINT
    "2 ADD TO TEXTFILE": PRINT
33  PRINT "3 PRINT ALL RECORDS": PRINT : PRINT
    "4 PRINT SELECTED RECORD": PRINT
34  PRINT "5 EXIT PROGRAM   ": PRINT
35  INPUT "SELECT BY NUMBER ";S
36  S% = S
37  IF S% < 1 OR S% > 5 THEN 31
38  ON S% GOTO 40,110,300,500,9999
105  GOTO 31
230  GOTO 31
470  GOTO 31
580  GOTO 31
```

SAVE as RAN-ACC CREATE

RUN — select various options and input.

Conclusion

You should now be able to write your own programs to enable you to work with textfiles.



18 Word processing

This chapter is to be used with the word processing program 'Zardax', produced by Computer Solutions, Queensland. It is designed as a self-paced instruction package to give an understanding of the word processing or text editing capabilities of the Apple //e when used in conjunction with the 'Zardax' program. Check the manual accompanying the disk for details of slight modifications needed to allow the word processing disk to operate.

You will also need another disk on which to save your documents. This disk will be referred to as the **document disk**.

When typing reaches the right hand side of the screen the next letter will appear on the next line. No action need be taken as the system will format the document when it is being printed. The program will produce a standard format unless you make adjustments. This will be covered later in the chapter.

What is word processing?

Word processing is the simple process of getting thoughts on paper and actioned. It may be defined as the process of production and distribution of text, eg letters, reports, lists, etc and it has come to mean the use of equipment to assist this process.

Stages of word processing

1. Initial input of ideas through dictation, shorthand or longhand.
2. Typing, drafting, revision — where text is produced in clearly readable form, perhaps after undergoing revision. This is commonly referred to as **text editing.**
3. Distribution of final copy.

Starting the system

1 Insert the Zardax master disk in Drive 1. Turn machine on and close disk drive door.

2 When the red light goes off, the screen display will give you a choice of:
PRESS S FOR SET UP or PRESS ANY OTHER KEY

Press any key except S.

3 The main menu will now appear on the screen, and this will list the **options** available.

The most frequently used of these options will be covered in this chapter:

Create, Print, Retrieve, Delete, Glossary, Newdisk.

4 Remove the Zardax master disk and insert the document disk in Drive 1. This will be used to store letters and other documents. Close drive door.

5 If the document disk has not been used before it will need to be initialised (see chapter 8 for meaning of this term).

To initialise, press N for Newdisk.

The initialising process is used also to wipe a previously recorded disk, so be very careful when using this process. If the disk you are using is full, initialise it to wipe previously recorded documents. To guard against accidentally wiping a disk it is wise to 'write-protect' the disk. This warning will appear on the screen.

6 To initialise press *. (*Remember:* SHIFT – 8.) This process takes about 60–90 seconds, depending on the system.

7 After initialising, the main menu will again appear on the screen, together with the number of sectors available on the disk. An unused disk contains approximately 528 sectors. The screen display of the number of sectors available will assist in estimating the amount of space available on the disk.

Learning about the options

To select any option, simply type the first letter of the word.

Create a new document — title: DOC 1

1 From the main menu choose Create by pressing C.

2 Name the document — type DOC1 . You may use up to 8 characters. If you don't use the maximum number then you must press RETURN to move to the next step. If you use more than 8 characters it will stop printing at 8 and will return to the next instruction.

3 Enter notes. You could combine the author's initials and data, or any other relevant details — maximum 11 characters.

4 The document name, author's initials and/or notes will appear at the

bottom of the screen, together with a number which indicates the position of the character in the paragraph.

5 Use SHIFT key normally in your typing. To type totally in capitals (upper case) press the CAPS LOCK key once — to release simply press CAPS LOCK again.

Use DEL to backspace to make corrections — this will remove the character before the cursor.

6 If no format instructions are given, Zardax has an assumed format for printing. Format instructions will be covered later in the chapter. The following exercise will use the assumed format.

Now type the following:

Large companies have traditionally used computers to solve most of their business problems. Small businesses must also cope with the increasing complexity of information and the need for providing immediate information about their business and the extremely vital need to control business costs.

The microcomputer is a system that the clerk-typist can use quite easily. It can be placed anywhere an office typewriter would normally be stationed.

Saving on disk

The document is in the memory of the computer. Should there be a power failure it would be lost. If you wish to save the document for future reference it is essential to save it on disk.

1 Press ESC (Escape key) to display the inner menu. The document will disappear from the screen but is still in the computer's memory.

The following options will appear on the screen:

Change
Draft
Main Menu
Print
Rename
Save
Videoprint

Which?

2 The inner menu lists options which refer to the document on which you are currently working. To select any option, simply type the first letter of the word.

3 Press S for Save — the red light will come on for a few seconds as the document is recorded on the disk.

Making changes

This document is still in the memory of the computer until you return to the main menu. To make changes or additions to this document now, call up Change from the inner menu — C. Previous typing will reappear on the screen to allow you to make changes. Notice that the cursor is positioned at the end of your previous document.

Make the following addition to DOC1:

The microcomputer brings a new age to office automation. The small businessman now does not have to make his office fit the traditional computer system, ie a separate central processor, mass storage device, printer or video display unit, or purchase expensive furniture to house this type of system.

The integration of hardware and software offers an outstanding total computer system for the small businessman.

Note: What you have on disk is now different from what is in the memory of the computer.

Save the new document — DOC1

Remember: ESC — to return to inner menu

S — to save

The old document is wiped from the disk and the new document replaces it.

The new document is now saved with the extra paragraphs added.

Return to main menu — M — DOC1 is now wiped from the computer's memory but is stored on disk for future reference.

Assumed format for printing

Unless you establish commands for printing, Zardax will assume the following format:

Left margin set at zero
Right margin set at 65
Single line spacing
Length of paper 66 lines
Typing to finish at line 54
No page numbering
No justification
Pitch 10
Continuous stationery

Formatting a document

To format a document means to establish commands for the printer — margins, line spacing, pitch, tab stops, page length, length of document, continuous stationery or separate sheets.

The basic command to the printer is APPLE-Ø which actually prints . Be sure to press and hold the APPLE and then press Ø. You may use *either* APPLE key.

Left margin: APPLE–ØLM followed by number	_LM15
Right margin: APPLE–ØRM followed by number	_RM65
Pitch: choice of 10 or 12 pitch APPLE–ØPI followed by number	_PI12
Line spacing: choice of	
single—APPLE–ØSS	_SS
double—APPLE–ØDS	_DS
1.1/2—APPLE–ØSH	_SH
Paper length: A4 sheet normally 66 lines APPLE–ØFL followed by number	_FL66
Text to end on line: APPLE–ØPL followed by number	_PL60
Justify the right margin, ie to finish all lines on right margin: APPLE–ØJU-	_JU
To stop justification of text: APPLE–ØNJ-	_NJ
Text to be printed on separate sheets: APPLE–ØCS-	_CS

Create a new document—title: DOC2

Set the following format for DOC2 (there is no need to space or return between format instructions):
Left margin 10; right margin 70; pitch 12; single line spacing; length of paper 66; printing to end 60; print on separate sheets; no justify

Check format instructions:

_LM10 _RM70 _PI12 _SS _FL66 _PL60 _CS _NJ (RETURN)

Centring a heading

APPLE–ØCE—type heading and then press RETURN.
Press APPLE–ØNC (to remove centering instruction).
Now centre the following heading in block capitals—use CAPS LOCK:
WIN THE PAPER WAR
Check: _CE WIN THE PAPER WAR
_NC

Now type the following document:

> Many small businesses today are staggering under an avalanche of paperwork. With spiralling wages it is harder and harder to keep costs down while managing the necessary workload. New technology has brought computers into the reach of most small businesses, allowing them to reap the benefits previously afforded only to large organisations.

The use of pre-packaged software is a large saving to small businesses because the cost of writing the program is shared among the many people who use it. As the programs have been previously tested, valuable time is saved.

Save this document on disk.

Add to the document—DOC2

(From inner menu call up Change. This calls up the document from the computer's memory not from the disk.)

Remember: To centre the heading: APPLE–ØCE—type heading
APPLE–ØNC

DON'T BE A LOSER

Your business could be losing money through inefficient collection procedures. Overdue accounts tie up valuable working capital and if not followed up can result in bad debts.

(Now change format to LM20 RM60 DSJU.)

The reports in this program can save time and money by showing you which accounts are overdue and how long they have been outstanding. You can write your own standard collection letters which will be produced by the computer and will incorporate all relevant information.

(Now reset format to original instructions.)

At the end of the month statements can be generated automatically, thus saving time and costly mistakes.

Renaming a document—title: NEWNAME

Return to inner menu (ESC).

Call up the Rename option by pressing R. Rename this document NEWNAME so you can easily identify it in the main menu.

You have only renamed the document which is *in memory*—you have not yet saved the renamed document on the disk.

From the inner menu press S (for save)—the document NEWNAME will then be saved on the disk.

Printing copy from memory

In the format you indicated that printing would be on separate sheets (cut sheets). If this document is too long for one page, printing will stop. Insert paper in printer and follow instructions from the screen.

From the inner menu print a copy of this document.

1 Call up Print by pressing P.

2 Turn printer on and set paper to correct starting position.

3 You will be asked how many copies you want before printing will start.

To stop printing

Use either ESC or CTRL–RESET.

Printer may not stop immediately—the computer will immediately stop sending further characters, but the printer will continue to print until it finishes all those characters which it has already received and stored.

Reprint

During printing, if you press R (Reprint) the computer will stop sending characters to the printer until you press any other key. It will start reprinting again at the beginning of the document—useful, especially if paper becomes jammed.

Returning to main menu

Remember that the document is now cancelled from the computer's memory, and unless you had saved it on disk it would be lost. If there are more than 21 files on the disk, press SPACE BAR to see subsequent entries in the catalog.

Main menu on screen will display programs saved on disk:

DOC1
DOC2
NEWNAME

} Each program name will be preceded by a code, eg A1, A2, etc

Printing copy from disk

As the main menu controls what is saved on disk, call up Print by pressing P.
Identify document by code.
Turn printer on.
How many copies?—indicate by number.
Press RETURN.

Print a copy of DOC2 and NEWNAME

Check the format instructions and note the additional information in the NEWNAME document.

Editing

To make changes to a document you must first retrieve it from the disk and transfer it to memory.

1 From main menu press R for Retrieve.

2 Enter the code for the document—DOC2.

3 The document will appear on the screen. The cursor will appear at the end of the document.

Pure cursor moves

Press and hold APPLE and then press letter:

APPLE–B moves cursor to beginning of text
APPLE–E moves cursor to end of text
APPLE–R moves cursor one space to right (or press RIGHT ARROW)
APPLE–L moves cursor one space to left (or press LEFT ARROW)
APPLE–U moves cursor one line space up and to beginning of line (or press UP ARROW)
APPLE–D moves cursor one line space down and to beginning of line (or press DOWN ARROW)

Moving cursor quickly

1 Hold directional arrow down to engage automatic repeat function.

2 APPLE– ↑ moves cursor up 10 lines.

3 APPLE– ↓ moves cursor down 10 lines.

Practise moving the cursor in various directions.

Insertions

Position the cursor at the beginning of the word where you wish to insert data and type. The information is automatically added.

Deletions

To delete one character at a time use DEL or APPLE–DEL. DEL deletes the character before the cursor; APPLE–DEL deletes the character at the cursor.

These keys may be held down to engage the repeat function.

Making corrections to DOC2

Is your business in this category?

Many small businesses today are staggering under an avalanche of paperwork. With spiralling wages it is harder and harder to keep costs down while managing the ~~necessary~~ ever-increasing workload. New technology has brought computers into the reach of most small businesses allowing them to reap the benefits previously afforded only ~~to~~ the large organisations.

delete

The use of pre-packaged software is ~~a large~~ an immense saving to small businesses because the cost of writing the program is shared among the many ~~people who use it~~ users. As the programs have been previously ~~tested~~ trialled, valuable time is saved.

Remember: What you have on the screen is now different from what is stored on disk.

Save the new document under the same name—DOC2.

Print a copy of the DOC2 from disk (remember to return to the main menu).

Retrieve the document titled NEWNAME.

Make the following corrections to NEWNAME.

Your business could be losing money through inefficient collection procedures. Any overdue accounts tie up valuable working capital and if not followed up by you can result in bad debts being incurred.

The reports in these ~~this~~ programs can save both time and money by showing you which accounts are overdue and how long they have been outstanding.

You can write your own standard collection letters and these ~~which~~ will be produced by the computer automatically and will incorporate all relevant information.

At the end of the month statements can be generated automatically, thus saving time and costly mistakes.

Save the amended document on disk.

Deleting a document from disk

Return to the main menu.
Select D for Delete option.
Insert the code number for DOC1.
Document DOC1 will be deleted from the disk.

Create a new document—title: TRAIN1

Set format: LM10 RM70 DS

Type the following:

The increasing use of computer technology not only influences the way that businesses and organisations conduct their affairs, but also the way we live. The impact on people has been dramatic.

A critical issue is whether or not the total number of jobs is increased or decreased by this technological innovation.

The basic structure of employment has changed, as many existing jobs have altered or been eliminated.

New jobs have been created in computer-allied industries, such as retail outlets for computers and technicians to service machines.

Underscoring

Position cursor at beginning of word.
To underscore one character—APPLE–Z.
To underscore whole word—hold APPLE and press Z for the number of characters to be underscored.

To remove underscoring

To remove underscoring from one character—APPLE–Y.
Hold APPLE and press Y to cancel underscoring for the whole word.

Practise underscoring words and removing underscoring.

Moving paragraphs

1. Position cursor anywhere within the paragraph to be moved.
2. Press APPLE–M to enter Move mode.
3. MOVE U/D will appear on the screen. (U = UP D = DOWN)
4. On pressing U or D the paragraph will move up or down a line. Press U or D again and it will jump the whole paragraph above or below it.
5. Press any other key to come out of edit mode.
6. Now insert the extra RETURNS.

Move paragraph 3 up to become paragraph 2.
Move paragraph 3 down to become paragraph 4.

Deletions

To delete larger sections of text—APPLE–W.
"Wipe What? P/A/B" will appear on screen.
P—present paragraph—deletes from cursor to end of paragraph. If cursor is at beginning of paragraph it will delete whole paragraph.
A—deletes everything above cursor.
B—deletes everything below cursor.

If you decide not to delete anything when you have entered delete mode, press any key other than P, A or B to escape.

Practise deleting sections of your document.

Save this document on disk.

Using the tabulator

To move to tab position

APPLE-T or TAB key  —moves cursor to preset tab stops
Set every 10 spaces with the first tab on 12

To clear tab

APPLE–T or TAB key—moves cursor to tab stop
APPLE–C—clears tab stop

To set new tab

APPLE–S—sets new tab

Create a new document — title: TAB 1
Centre this heading:

TAB EXERCISE 1

Using pre-set tabs, enter:

5	10	15	20	25	30
40	50	60	70	80	90

Now clear all tabs, and set new tabs at:

5	17	29	41	53	65

Using these tabs, enter the same numbers as in your first table.
Save document on disk.
Print TAB1 from disk.

Create a new document — title: TRAIN2
Type the following:

In an effort to ensure award coverage for all public sector professional engineers in Tasmania the Association has commenced proceedings to provide award coverage for the professional engineers employed by the Port Authority Tasmania and Transport Commission, Tasmania.

This will then give the engineers concerned the opportunity to take matters such as reclassification to the Arbitrator in Tasmania.

Find and replace (search and substitute)

Sometimes you may want to search for a particular word or group of words in a document and substitute another word in its place (eg month, surname, town).

1 Press APPLE–B to return cursor to beginning of document.

2 Press APPLE–F (Find).

3 FIND: ? Type the word with a space *before* and *after*.
Type engineers (RETURN)
(If searching for 'the' and no space is included before and after, the computer will select all 'the' combinations, even in the middle of a word.)

4 REPLACE WITH: ? Type replacement word with space *before* and *after*.
Type surveyors (RETURN)

5 The cursor will move to the first word it was to FIND in the text.

6 As the first word is located you will be asked 'Y/N/A'
Y = Yes N = No A = All

If you press A, every time this combination appears it will be deleted and the new word/s will be substituted. Otherwise you choose at each occurrence whether you wish that particular word to be cancelled. If you press N at one particular word, the cursor will move to the next location of the word for which you are searching.

To escape from this mode press any key other than Y, N or A.

Practise this by making the following substitutions in TRAIN2:

1 Replace 'award coverage' with 'correct salaries'.

2 Replace 'Tasmania' with 'Queensland'.

Did you notice that only two words were changed? The other two words had a full stop following the word. The system will find only the exact word or words you have requested.

Let's revise other editing features

1 Move paragraph 2 to become the first paragraph.

2 Underscore three separate words in the second paragraph.

3 Delete the first paragraph.

4 Insert the following sentence at the end of the paragraph:
The procedure involved requires that each Port Authority be declared a controlling authority.

5 Save the document on disk.

Standard paragraphs

Create each of the following standard paragraphs as a separate document, titled PARA1, PARA2 and PARA3.

PARA1:

We wish to bring to your notice the fact that the above account is now overdue. We feel sure that this must be an oversight on your part and look forward to receiving a cheque within the next few days.

Save on disk

PARA2:

May we draw your attention to the fact that the above account is now overdue. Settlement by return mail would be appreciated.

Save on disk.

PARA3:

As we have received no reply to our previous letters regarding this account, we regret to advise that the matter has now been placed in the hands of our solicitors, Messrs Jones & Associates, and you will be liable for the costs incurred.

Save on disk.

Merging documents — using standard paragraphs to create a new document

It is possible to merge several documents.

Create a new document — title: INSERT

Type the following paragraph:

An important task required in word processing is combining standard paragraphs to make a new document.

Follow these steps:

1. Position cursor where new paragraph is to be inserted. Press RETURN twice.
2. Press APPLE–I (Insert).
3. From list of documents saved on disk choose code beside PARA2.
4. Paragraph will appear where cursor was positioned.
5. RETURN twice for new paragraph and type the following:

   Each standard paragraph should be composed as a separate document.

   RETURN to indicate new paragraph.

6. Call up PARA3.
   RETURN for new paragraph.
7. Call up PARA1.

Save this document on disk.

Practise printing a copy of this document from disk from the main menu.

Glossary

If using phrases or paragraphs regularly you can create a **glossary** which will reproduce these phrases quickly. Each glossary may have a maximum of 26 phrases which are coded by the letters of the alphabet.

Creating a glossary item

To indicate the beginning of the first phrase:

APPLE–Ø twice __ (indicates beginning)
Type letter b (code letter for this reference)
Type glossary entry
APPLE–Ø 3 times ___ (indicates end)

eg __bDear Sir___

Create a new document — title: GLOSS 1

Create the following items in this glossary:

Format — a — left margin 15; right margin 65; length of page 66 lines; finish printing text on line 60; pitch 10; single line spacing.

eg __a_LM15 _RM65 _FL66 _PL60 _PI10 _SS ___
Salutation — b — Dear Sir
eg __bDear Sir___
Complimentary close — c — Yours faithfully
4/5
Manager

Standard paragraph 1 — d — We wish to bring to your notice the fact that the above account is now overdue. We feel sure that this must be an oversight on your part and look forward to receiving a cheque within the next few days.
Standard paragraph 2 — e — May we draw your attention to the fact that the above is now overdue. Settlement by return mail would be appreciated.
Standard paragraph 3 — f — As we have received no reply to our previous letters regarding this account, we regret to advise that the matter has now been placed in the hands of our solicitors, Messrs Jones & Associates and you will be liable for the costs incurred.
Standard paragraph 4 — g — May we draw your attention to the above account, which has now extended considerably beyond our trading terms. We regret to advise that until this account is settled no further credit can be allowed.

Save this document on disk.
Return to main menu.

How to use a glossary

1 Load glossary into the computer's memory — press G.

2 Screen will display Which Document? Use the code beside GLOSS1.

This glossary is now loaded into the memory and can be called up for use until a further glossary is loaded.

Create a new document to use this glossary — title: LETTER1

When you wish to take any item from the glossary, use APPLE–G.

Now create the following form letter. Use full block style with open punctuation.

1 Recall Format — APPLE–Ga — and press RETURN.

2 Type date and RETURN 3-5 times

3 Type inside name and address:
Mr A Smith
11 Far Road
BRISBANE 4000

4 Recall Salutation — APPLE–Gb — RETURN twice.

5 Recall Standard paragraph 4 — APPLE–Gg — RETURN twice.

6 Recall Complimentary close — APPLE–Gc — RETURN twice.

7 Add your reference initials.

To print copy from inner menu

As this is a standard letter there is no need to save it on disk. Copy will be printed from the memory only.

Personalising a form letter

Create a new document—title: LETTER2

When sending the sample letter to numerous people (ie form letter), you can personalise it. Before printing the letter, you will be asked to supply relevant details either from the keyboard or from a prepared file of mailing addresses. Provision is made for this in the draft of the letter. The curly brackets are used to show where information can be inserted in the final printing.

1 Recall Format (and press RETURN).

2 Type date

3 {Name}

4 {Street}

5 {City} {Postcode}
(If state required, insert in brackets.)

6 Recall Salutation.

7 Recall Standard paragraph 2.

8 Recall Complimentary close.

9 Type reference initials.

Print copy from Draft on the inner menu. It will print exactly as shown on the screen—with curly brackets. This can be kept as a sample.

Print copy from Print on the inner menu. Choose from the screen whether the information is to be entered from diskette or keyboard. Choose keyboard. You will be asked to type information before printing begins.

Addressee 1 Mr A Person
123 Collins Street
MELBOURNE 3000

Addressee 2 Ms J Johnson
24 Pitt Street
SYDNEY 2000

As these are personalised letters, there is no need to save them on disk.

Return to main menu.

Additional print commands

Try these commands on the documents you have saved. Remember that print commands can be used within a document to change format.

_MA 10 — sets an indentation from the left margin of ten spaces

_MAØ — removes the margin command

_IN20 — does not affect the first line of typing, but indents subsequent lines by twenty spaces from the left margin

_MAØ — removes the indent command

_PN1.40 —for automatic page numbering at top of page. In this example 1 = page number 1, and 40 = the position across the page where the number will be printed. Note that page 1 will not be numbered, but all subsequent pages will be.

_NN — removes the page numbering command

_HD — defines a **header** (ie constant message to be printed at the top of each page except the first)

eg _HD
HEADING
(2 blank lines)
APPLE-Ø

_HO — Header on. After this command the header will be printed automatically at the top of each page (except the first)

_HN — removes 'header on' command

_FD — defines a **footer** (ie constant message to be printed at bottom of each page, or for section titles)

eg _FD
(2 blank lines)
FOOTER
(1 blank line)
APPLE-Ø

_FO — Footer on. After this command the footer will automatically be printed at the bottom of each page

_FN — removes 'footer on' command

_NP — New page. Next output will be printed on the next page.

Numbered indented paragraphs

Follow this example. Make sure that Justify is off.

10.APPLE–ØIN6 This is an example of numbered indented paragraphs. The number is printed at the left margin, but all following text is indented six spaces. Remove this command with APPLE–ØMAØ.

This example would print as:

10. This is an example of numbered indented paragraphs. The number is printed at the left margin, but all following text is indented six spaces. Remove this command with APPLE–ØMAØ.

In conclusion

The main options available in this word processing system have been covered, but there are further options available which we have not covered. Some of these are outlined below:

Lock—locks a document on the disk.

Unlock—unlocks a document on the disk.

Transfer—transfers document to another disk.

Index—prints the catalog on the printer.

You should now be able to apply the basic principles to cover any further options with which you wish to experiment.

Index

Shake hands with the Apple//e has been designed as an individualised programme to help you learn the operation of the Apple//e microcomputer.

Operating instructions relate specifically to the Apple//e, but in most cases they will also apply to other versions of Apple series II microcomputers.

Shake hands with the Apple//e is written in an easy-to-understand style which assumes no knowledge of computers or computing jargon. It takes you from simple programs and programming to graphics and word processing.

Pitman

ISBN 0 85896 044 3