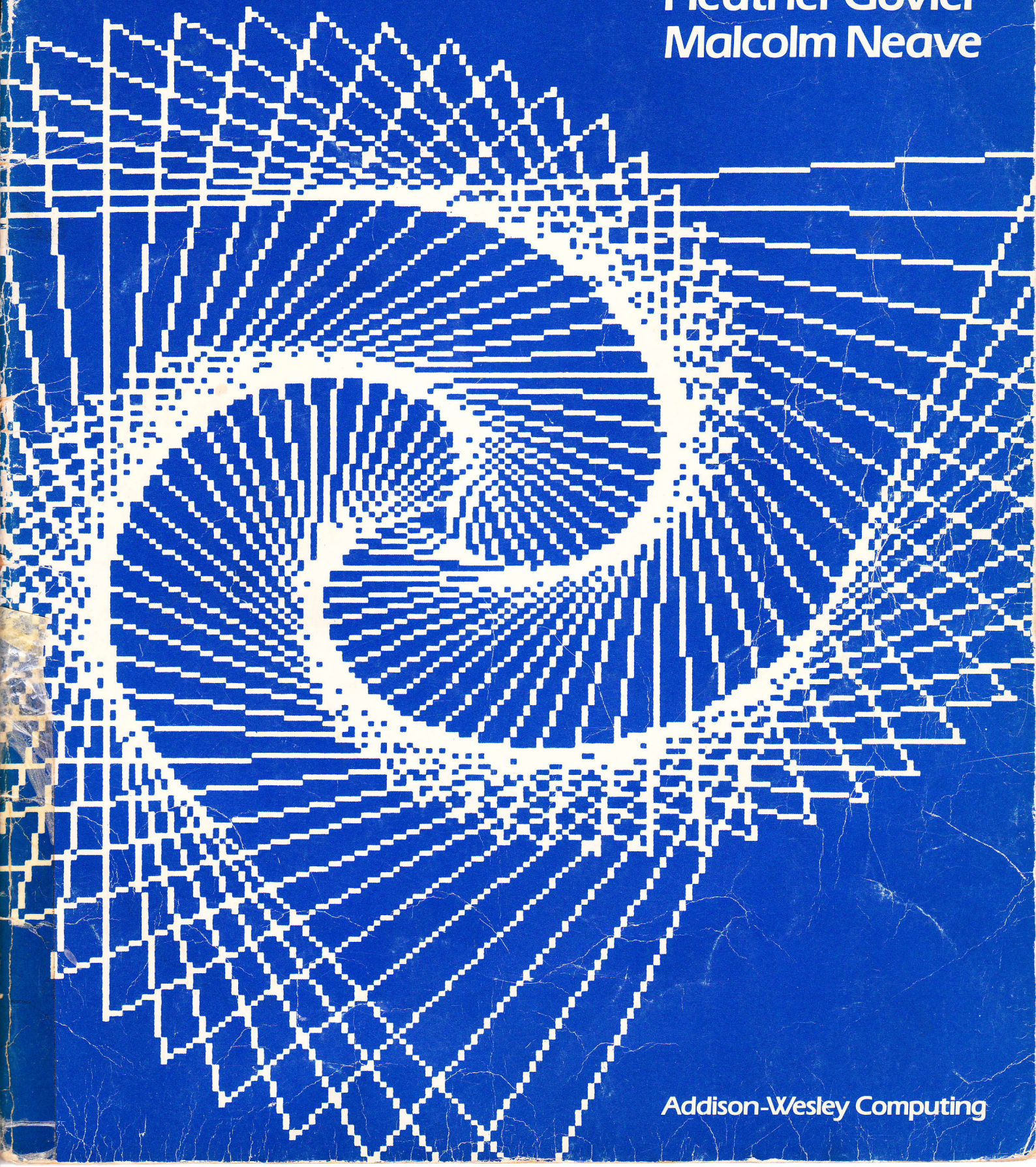
LOGO CHALLENGE

Teacher's Guide

Heather Govier
Malcolm Neave



Addison-Wesley Computing

Important notice

Please read carefully the following information.

This program is protected by International Copyright Laws. Any unauthorised copying is illegal.

Addison-Wesley provides comprehensive support services, as follows.

Replacement copy service
Should your software be faulty, please return it to us within 21 days of receipt, indicating the nature of the fault plus the date of order and invoice number.

Accidental damage after purchase
In the event of accidental damage during use, you can purchase replacement software. Please write, stating the version to be replaced and quoting the ISBN and Serial Number printed on the label of your original cassette or disk.

For further information on Addison-Wesley computing books and software, please contact your local Addison-Wesley office or,

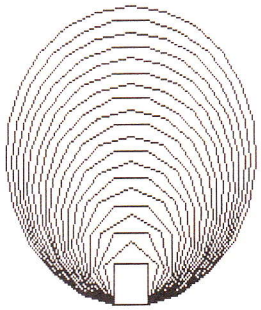
Addison-Wesley Publishers
53 Bedford Square
London WC1B 3DZ
Telephone 01-631 1636

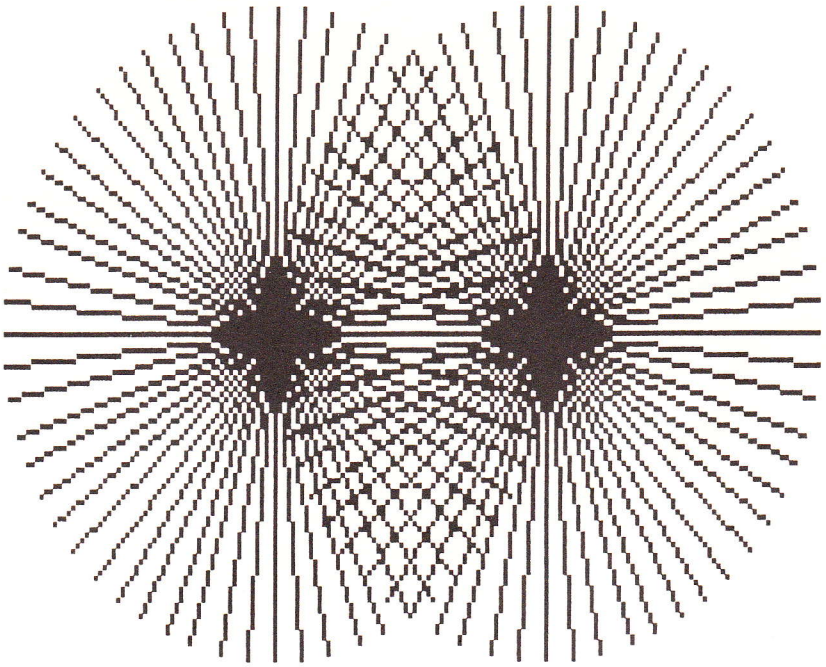
LOGO CHALLENGE *Materials available:*

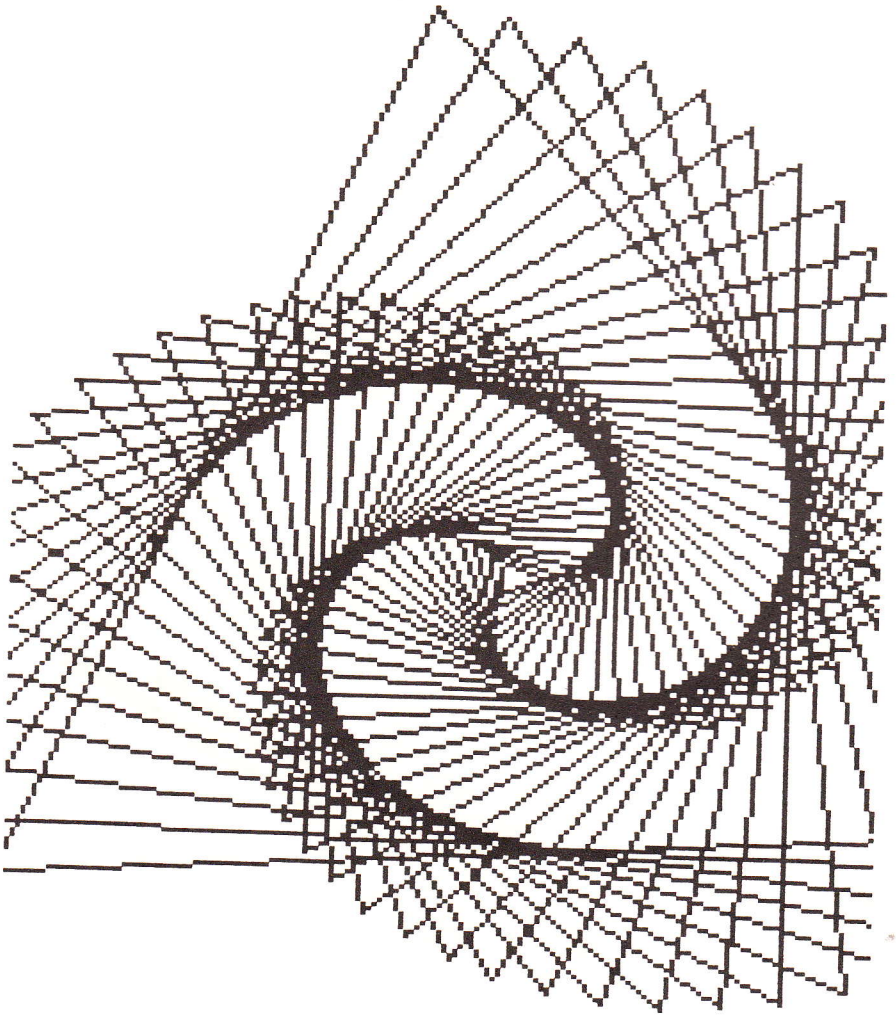
Teaching Pack for BBC Acorn Model B	201 18401 X
Teaching Pack for Sinclair ZX Spectrum	201 18404 4
Teaching Pack for Research Machines 480Z	201 18402 8
Teaching Pack for Research Machines 380Z	201 18403 6
Pupil Pack (5 copies of Pupil Book)	201 18400 1

LOGO CHALLENGE

Teacher's Guide





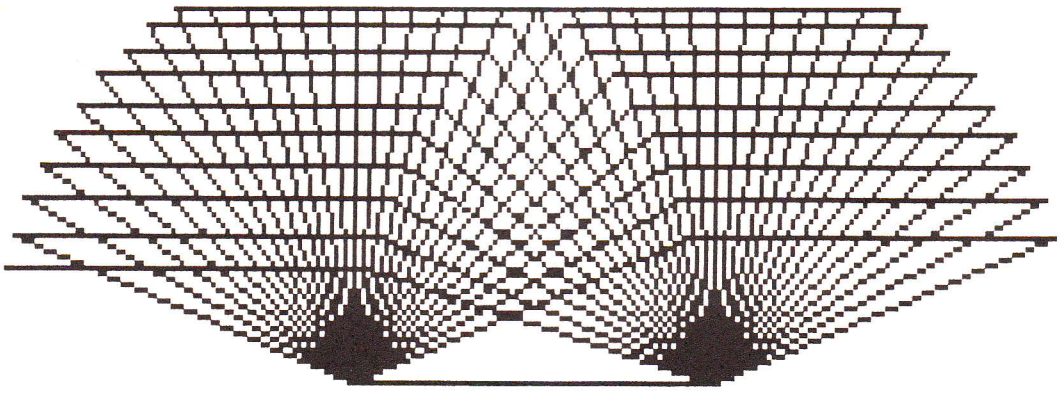


LOGO CHALLENGE

Teacher's Guide

Heather Govier Malcolm Neave

Adviser Paul McGee



Addison-Wesley Publishers
London • Reading, Massachusetts • Don Mills, Ontario • Menlo Park, California • Amsterdam
Manila • Sydney • Singapore • Tokyo

Acknowledgement

The authors and publishers wish to thank Research Machines for permission to use RML Basic 5·0A in this project.

The authors wish to thank colleagues at the Davidson Professional Centre, Croydon for their help and advice in developing this project; and all the teachers and pupils who participated in school trials.

Addison-Wesley Computing

LOGO CHALLENGE

© 1983 Addison-Wesley Publishers Limited
53 Bedford Square, London WC1B 3DZ

All rights reserved. No part of this publication, including computer software, may be reproduced , stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without prior written permission of the publisher.

ISBN 201 18410 9

Design and illustrations by Sharp Print Management

Cover design by Jeff Carter

Printed in Great Britain by Thomson Litho Ltd, Scotland

ABCDEF 89876543

CONTENTS

Section One	INTRODUCTION	1
	Computer Software Pupil Book Teacher's Guide Background	
Section Two	TEACH YOURSELF LOGO CHALLENGE: PART 1	3
	Starting LOGO CHALLENGE commands	
Section Three	TEACH YOURSELF LOGO CHALLENGE: PART 2	8
	Variables More about EDIT Arithmetic with variables	
Section Four	SAVING YOUR WORK	12
Section Five	USING LOGO CHALLENGE WITH CHILDREN	13
	A note about LOGO Organisation Introducing LOGO CHALLENGE Planning programs Storing programs Combining procedures Introducing variables and editing Drawing circles Variables with arithmetic The challenge of LOGO CHALLENGE	
Section Six	NOTES ON CHALLENGES IN THE PUPIL BOOK	16
Section Seven	THE ANSWER FILE	23
	Loading Uses Storing extra Challenges	
	GLOSSARY OF LOGO CHALLENGE COMMANDS	24
	TECHNICAL SPECIFICATION	25

This project is dedicated to Seymour Papert
whose book, *Mindstorms,* was the catalyst of our work.

Section One

INTRODUCTION

LOGO CHALLENGE is a computer program for children. It has been designed to promote problem solving and logical thinking and can be used with pupils from a wide range of age and ability levels. The program is aimed at children of average or above average ability between the ages of nine and thirteen years, but, with guidance, can also be used with younger children.

Through LOGO CHALLENGE, children are introduced to the logical problem-solving skills involved in programming a computer. The child's role is to command a "screen pen" to draw pictures or patterns. At first these may be simple shapes such as squares or triangles. Later these basic elements can be combined to produce complex and fascinating constructions.

Even at the simplest level there is a need for clear thinking and logical analysis. Only a carefully thought out and precise series of instructions will produce the required result. Anything else will result in a flaw in the picture or a "bug" in the program. The appearance of such a bug presents the learner with a challenge to find and correct the error.

LOGO CHALLENGE comprises:

1 *Computer Software* - LOGO CHALLENGE program and Answer File
2 *Pupil Book* - Lessons and Challenges
3 *Teacher's Guide*

Computer Software

The program is available on cassette or disk for the following microcomputers:

BBC Acorn Model B - cassette
RML 480Z - cassette
RML 380Z - disk
Sinclair ZX Spectrum - cassette

In addition to the LOGO CHALLENGE program, on each cassette or disk is an Answer File. This file contains procedures to draw all the pictures set as Challenges in the *Pupil Book,* and a number of other interesting items. (See page 23 of this book.)

Pupil Book

This is structured as a series of Lessons and Challenges, requiring increasing levels of thought and planning. LOGO CHALLENGE commands are introduced systematically in the Lessons, and the Challenges linked to each Lesson provide opportunities for pupils to experiment with each new set of commands. The Lessons and Challenges are progressive, building up slowly until pupils are able to produce complex and sophisticated patterns. For the more able student, the *Pupil Book* should be self-sufficient and there will be little need for teacher intervention, especially in the early stages. Other pupils may need more help. In order to give adequate support to those children who require it, teachers must be familiar with the program.

Teacher's Guide

This is designed to give teachers the necessary background information for them to be able to provide the support mentioned above. In a demanding situation, where reference to an answer book is not possible, confidence can only come from familiarity with the program. The *Teacher's Guide* is thus a vital component of this package.

Sections Two and Three of the *Guide* are called Teach yourself LOGO CHALLENGE: Parts 1 and 2. In Part 1 all the LOGO CHALLENGE commands are explained and opportunities provided to practise their use. Part Two goes into specific aspects in more detail, enabling the drawing of more complex patterns and shapes.

A teach yourself format has been adopted, and Sections Two and Three should be worked through thoroughly at the keyboard before introducing the program to children. While doing this it will be helpful to have the *Pupil Book* handy, both for the pictures of what should be on the screen and in order to experiment with the Challenges.

Section Six is a set of notes on the Challenges presented in the *Pupil Book.* These do not constitute a set of standard answers, but provide useful hints and approaches to tackling the problems and can be readily referred to during a lesson. Standard answers are not provided because, in all work with LOGO CHALLENGE, it is not the solution itself which is important but the thinking process required to attain it. Indeed there is no one "right" answer to any of the Challenges. Many alternative solutions are possible.

Background

The program LOGO CHALLENGE is based on the computer language LOGO, developed in America in

the late sixties. This language was first used with children by Seymour Papert at the Massachusetts Institute of Technology. He describes his experiences in a very stimulating and exciting book called *Mindstorms.* This book not only describes the children's learning but also sets forth a philosophy of education which underlies all the work with LOGO.

Essentially, the computer is seen as a tool to enable the child to take control of his/her own learning. Papert coins the term "microworld" to describe a learning environment. In the right microworld, children learn thinking skills easily and naturally because they are provided with all the necessary ingredients in an ordered and structured manner. Thus learning by discovery is made more sure and more speedy. With LOGO children can acquire thinking skills which, Papert claims, can then be applied to a multitude of problems outside the LOGO microworld, such as juggling or physics.

Extensive research would be needed to validate these claims. Certainly this *Guide* is not the place for detailed theoretical exposition. For further explanation of the LOGO philosophy of education readers are referred to *Mindstorms - Children, Computers and Powerful Ideas* by Seymour Papert. (Harvester Press, Brighton, 1981.)

Perhaps the most important tenet of this philosophy is that learning should be fun. This is true for adults as well as children. Teachers who have been introduced to LOGO CHALLENGE have almost without exception thoroughly enjoyed working with it. Because the tasks are open-ended, stimulus and challenge can be found at all levels. LOGO CHALLENGE is a computer program for teachers too.

Section Two

TEACH YOURSELF LOGO CHALLENGE: PART 1

Starting

Load the program from cassette or disk in the usual way. (In case of difficulty, see page 25.) The project title will be displayed. After the RETURN key is pressed, the program will continue by asking whether you have a data file that you wish to load. As you have no file yet, type N followed by RETURN to start the program.

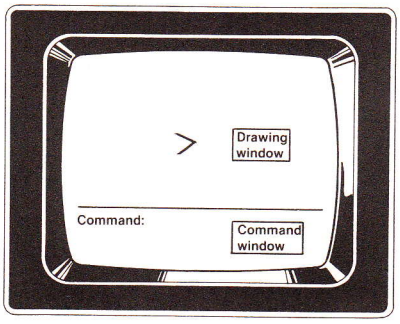
> **Note:** Throughout, the RETURN key must be pressed after each input.

After a short delay the following elements will appear on the screen.

1 Approximately in the centre will be a small arrowhead. This marks the position of the "pen".

2 At the bottom of the screen is the cue "Command:". This is a request for you to supply one of a number of commands which will move the pen. The full list of commands is given in the Glossary on page 24. Commands may be given either by typing in the full word or the two-letter abbreviation.

It will be seen that the screen is divided into two parts. The upper part (blue on a colour monitor) is the "drawing window". All the pictures will be displayed here. The lower part is the "command window". Up to four lines of commands can be displayed here.



LOGO CHALLENGE commands

It is suggested that you try out each of the commands as you work through the following notes.

FORWARD (FD)

This command must be followed by a number, and will then move the pen forwards on the screen by that number of units. (All numbers must be less than 180.)

Type: FORWARD 20 ⟶

The pen will then move forwards, i.e. East, for 20 units, drawing a horizontal line on the screen.

CLEAR (CL)

This command will erase all drawing from the screen and reset the pen to the starting position. It should be used before trying each of the commands given below.

HOME (HM)

This command will also reset the pen to the starting position, but without clearing the existing drawing from the screen.

Type: CLEAR
FORWARD 20 ⟶
HOME ⟶

BACKWARD (BD)

Similar to FORWARD, this command will move the pen backwards for a given number of units.

Type: CLEAR
BACKWARD 20 ⟵

The pen will draw a horizontal line extending 20 units West from the origin.

UNDO

This command will undo only the last instruction entered.

Type: CLEAR
FORWARD 20
FORWARD 60
UNDO

⟶ - - - - - - - - - - - - -

This will leave a line 20 units long, UNDO having erased the 60-unit line. It is a useful command for correcting a mistake in the latest of a long series of instructions.

The UNDO command can be used to undo up to five instructions. In order to do this, the word UNDO must be typed five times. For example, type the following series of instructions and observe what happens:

```
CLEAR
FORWARD 10
FORWARD 10
FORWARD 10
FORWARD 10
FORWARD 10
UNDO
UNDO
UNDO
```

LEFT (LT) — RIGHT (RT)

These commands will turn the pen, on the spot, to the left or right through any specified angle. However, as there is no forward or backward movement, nothing will be drawn on the screen.

```
Type: CLEAR
      LEFT 90
```

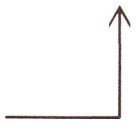
This will turn the pen 90 degrees to the left so that it will now face North.

```
Type: CLEAR
      RIGHT 90
```

This will reorient the pen to face South.

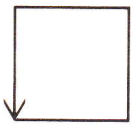
A BACKWARD/FORWARD command can now be given and the pen will move North or South. Thus the following will produce a right angle.

```
Type: CLEAR
      FORWARD 20
      LEFT 90
      FORWARD 20
```



Combinations of the FORWARD and LEFT 90 commands can be used to draw a square.

```
Type: CLEAR
      FORWARD 20
      LEFT 90
      FORWARD 20
      LEFT 90
      FORWARD 20
      LEFT 90
      FORWARD 20
```

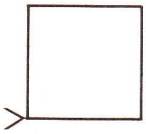


Square drawing thus takes 7 steps. A series of commands cannot exceed 120 steps, but such long-winded sets of instructions should not be necessary if optimum use is made of the next two pairs of commands. For example, the process of drawing a square can be shortened considerably, as follows.

REPEAT (RP) — AGAIN (AG)

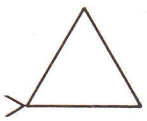
REPEAT followed by a number will cause the pen to execute that number of times any commands which follow. AGAIN closes the repeat loop.

```
Type: CLEAR
      REPEAT 4
      FORWARD 20
      LEFT 90
      AGAIN
```



Similarly to draw an equilateral triangle—

```
Type: CLEAR
      REPEAT 3
      FORWARD 20
      LEFT 120*
      AGAIN
```



> ***Note:** See page 13 for further discussion of triangle drawing.

It should now be possible to write a series of instructions to draw any regular polygon.

> **Note:** After typing the command REPEAT, the cue word at the bottom of the screen changes from "Command:" to "Repeat:". This is to remind the user that the command is in REPEAT mode. To get out of this mode, simply type AGAIN and the cue word will revert to "Command:". After typing all the lines to be repeated, it is important to remember to close the repeat loop with AGAIN. It is possible to get some very strange effects by forgetting to do this.

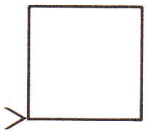
ESCAPE

It is possible to escape from a procedure at any time by using the ESCAPE key. This will stop the execution of any series of instructions and reset COMMAND mode.

DEFINE (DF) — END (EN)

Any set of instructions can be stored by use of the DEFINE mode, followed by a name. It can then be recalled by giving that name in response to "Command:".

```
Type: CLEAR
      DEFINE SQUARE
      REPEAT 4
      FORWARD 20
      LEFT 90
      AGAIN
      END
```



This will draw a square on the screen and also store this short procedure, which may be recalled at any time.

Clear the screen by typing:

```
CLEAR
```

and then respond to the next "Command:" cue by typing:

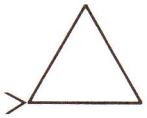
```
SQUARE
```

The square will be drawn as defined. Any other procedure may be defined and stored similarly.

A procedure may be given any name provided that the name contains only letters. Numbers and spaces are not permitted. So the SQUARE procedure above could be called "S" or "SQ" or

"ENGLEBERT", but not "ENGLEBERT HUMPERDINK" or "R2D2".

Type: CLEAR
DEFINE TRIANGLE
REPEAT 3
FORWARD 20
LEFT 120
AGAIN
END



Thereafter, typing:

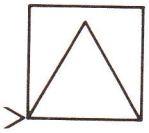
TRIANGLE

will call the TRIANGLE procedure. As with the REPEAT mode, when command enters the DEFINE mode after typing DEFINE, the cue word changes to "Define:". This, again, is simply as a reminder of what mode is in use. To get out of DEFINE mode always type END.

Note: Always remember to clear old drawings from the screen, by typing CLEAR, before starting something new.

The next step is to combine procedures. For example, SQUARE and TRIANGLE can be combined to form a house. Frequently the first response is to type:

SQUARE
TRIANGLE



Here is a bug. Consideration of the position of the pen at the end of SQUARE should help solve the problem. In order to draw a house it is necessary to move the pen between SQUARE and TRIANGLE. This can be done without leaving a trace, using the next pair of commands.

PENUP (PU) — PENDOWN (PD)

In order to move the pen without drawing a line on the screen, the pen should be lifted before it is moved. After a PENUP instruction the cursor will move without leaving a trace. Thus, after clearing the screen with:

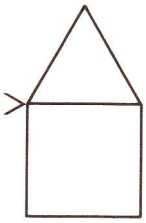
CLEAR

try the following instructions:

PENUP
FORWARD 20
PENDOWN ------>

This facility may be applied to the house-drawing problem.

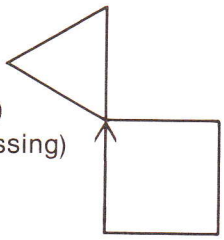
Type: SQUARE
PENUP
LEFT 90
FORWARD 20
RIGHT 90
PENDOWN
TRIANGLE



Eureka!

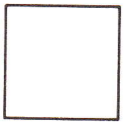
It may take some time to achieve this solution. Common bugs involve a failure to reorient the pen:

SQUARE
PENUP
LEFT 90
FORWARD 20
(RIGHT 90 missing)
PENDOWN
TRIANGLE



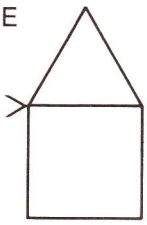
or failure to put the pen down again:

SQUARE
PENUP
LEFT 90
FORWARD 20
RIGHT 90
(PENDOWN missing)
TRIANGLE



The procedure HOUSE may now be defined in the same way as SQUARE or TRIANGLE.

Type: DEFINE HOUSE
SQUARE
PENUP
LEFT 90
FORWARD 20
RIGHT 90
PENDOWN
TRIANGLE
END



Thereafter, typing:

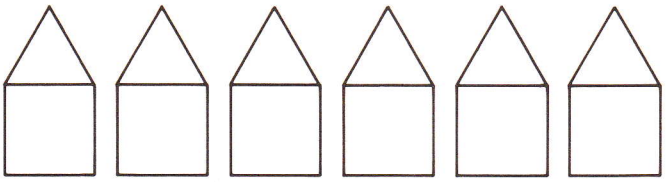
HOUSE

will produce the drawing as defined.

Note: Any defined procedure can be placed in position on the screen by first moving the pen using PENUP and PENDOWN to avoid leaving a trace.

Thus defined procedures can be nested inside one another. HOUSE is built up from SQUARE and TRIANGLE. Similarly STREET could be built up from HOUSE.

Type: PENUP
BACKWARD 150
PENDOWN
REPEAT 6
HOUSE
PENUP
FORWARD 30
RIGHT 90
FORWARD 20
LEFT 90
PENDOWN
AGAIN



This will produce a 6-house street. The first 3 commands reposition the pen so that there is plenty of room for all the houses. The commands bracketed together move the pen into the correct position to start the next house. Explore what happens if they are omitted. Use of DEFINE STREET . . . END stores the STREET procedure in a manner similar to before.

DESCRIBE (DS)

Any stored procedure can be called up in DESCRIBE mode. This will produce a step-by step description of the routine.

Type: DESCRIBE HOUSE

This will produce, at the foot of the screen, the word "SQUARE" which is the first line of this procedure. Use of the RETURN key will move through the description one line at a time. As the line-by-line description appears, the drawing will appear in matched stages in the upper screen.

LIST (LS)

This command will produce a listing of the names of all the procedures stored in the program. Thus typing the command:

LIST

at this juncture, will give: "SQUARE" "TRIANGLE" "HOUSE" "STREET".

LIST followed by the name of a procedure will produce a listing of the instructions used to define that procedure.

Type: LIST HOUSE

This differs from DESCRIBE in that there is no simultaneous screen display of the picture and thus the full listing can be seen at one time. This is very useful for reviewing and making changes to procedures.

DELETE (DL)

If DEFINE STREET is now typed, the reply will be "Already defined". Thus, in order to define a different street, it is necessary either to name it differently (e.g. ROAD or S) or to delete the previously defined procedure. This can easily be done by typing:

DELETE STREET

A listing will now give only SQUARE, TRIANGLE and HOUSE.

The program will not allow deletion of a procedure which is necessary for a more complex one.

Type: DELETE SQUARE

This results in the message: "I can't because it is used by HOUSE", because SQUARE is essential to HOUSE.

Note: As STREET is no longer stored in the memory of the computer, define STREET again and store it.

EDIT

It is possible to alter the instructions in a stored procedure using the EDIT command. For example, to alter the size of SQUARE, type:

EDIT SQUARE

This will produce a line-by-line description of the SQUARE procedure. Each successive line is viewed by pressing the RETURN key.

RETURN "REPEAT 4"

There is no need to change this, so:

RETURN "FORWARD 20"

This is the line to be edited to alter the size of SQUARE. Typing R will allow the line to be replaced by something else.

Type: R

This will produce "Replace:", which is a request for the new line to be input.

Type: FORWARD 50

Pressing the RETURN key now restores the line-by-line program description.

RETURN "LEFT 90"

There is no need to change this, so:

RETURN "AGAIN"

Further use of the RETURN key will result in a message: "Any more (Y/N)?", asking whether you wish to insert any further lines into the procedure. As you do not wish to do this, type N followed by RETURN, which restores the COMMAND mode.

Try editing SQUARE again, back to its smaller size.

Note: Further information on the use of the EDIT facility is given on page 9.

HIDE (HD) — SHOW (SH)

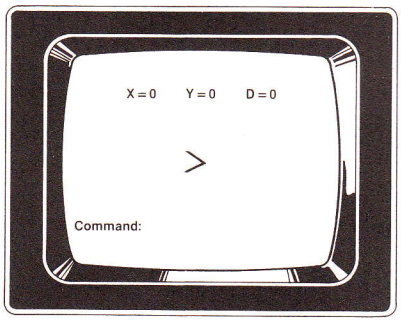
It is possible to hide the arrowhead before or after calling a procedure. Drawing is considerably faster when the arrow is hidden; thus, for more complex procedures where there is no need to monitor carefully the movements of the pen, it is as well to type HIDE first. Try this with the STREET procedure to see how much faster the drawing is completed.

Type: HIDE
STREET

To get the arrowhead back again simply type SHOW.

WHERE

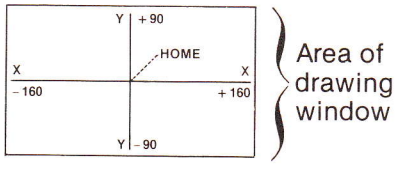
The current position of the arrowhead can be expressed in terms of X and Y coordinates, and the direction in which it is facing as an angle away from the starting position. Thus, typing WHERE, when the screen is blank after a CLEAR instruction, will produce the following, displayed at the top of the screen.



In accordance with standard notation, the starting direction (D = 0) is with the arrow pointing East, and the angle is incremented in an anti-clockwise direction. So LEFT 25 would produce D = 25, while RIGHT 25 (from HOME) would result in D = 335.

The HOME (centre-screen) position is at the intersection of the X and Y axes, so that positions to the left and below this will have negative coordinates, and those to the right and above, positive ones. The dimensions of the screen are:

X from −160 to +160
Y from −90 to +90



It is not likely that this facility will be used much with younger pupils, but for older students it can be valuable. To remove the X, Y, D, display, type WHERE again. This command thus acts as a *toggle,* the same command serving to switch the coordinates display on or off.

SETX — SETY

The position of the pen can be set to any position on the screen, with reference to the X and Y coordinates. This can be done at any stage, either before or during the course of a construction. SETX moves the pen across and SETY moves it up or down, to the given position.

Type: CLEAR
SETX −70
SETY 30

As these commands do not change the heading of the pen, they provide a useful way of moving the pen during the course of a construction. HOUSE, for example, could be defined with 4 fewer commands than before.

Type: SQUARE
SETY 20
TRIANGLE

SCALE

The SCALE facility has been incorporated to make the program easier to use with young children. It was felt that the larger numbers required to move the pen across the screen would probably be outside the experience of younger pupils. SCALE 10 can be used to increment the size of all movements by a factor of 10. So SCALE 10 followed by FORWARD 5 will produce the same movement that was produced previously by FORWARD 50. To return to the default position, reset by typing SCALE 1.

This command should not be used in the middle of a drawing as strange results could occur. It is suggested that it is used mainly to set the scale at the beginning of a work session. SCALE can be followed by any number less than 180.

HELP

The HELP file gives a listing of all the commands available in LOGO CHALLENGE.

Type: HELP

Further information about a specific command can be obtained by typing HELP followed by the name of that command. For example, HELP DESCRIBE will produce an explanation of the function of the command DESCRIBE.

FINISH

To terminate the program type FINISH. At this stage, any routine stored in the computer by use of DEFINE can also be stored on the cassette or disk. Details of how this is done are given on page 12.

Section Three

TEACH YOURSELF LOGO CHALLENGE: PART 2

Variables

All the procedures defined so far have included instructions which determined the size of the drawing. SQUARE, for example, included the instruction: FORWARD 20, which set the size of the square at 20 units. A more flexible procedure can be written if a variable instruction is given. Variables may be named as SIZE, ANGLE, NUMBER or VALUE. So:

```
FORWARD 20
```

can be replaced with:

```
FORWARD SIZE
```

Thus, to define a variable square (called VSQUARE) type:

```
DEFINE VSQUARE
REPEAT 4
FORWARD SIZE
```

At this juncture the reply will come: "SIZE not set—how big should it be?" Any value can be typed, e.g. 20. This will not be integral to the procedure VSQUARE but, as the rest of the definition is typed, a SIZE 20 square will be drawn. After typing 20, type:

```
LEFT 90
AGAIN
END
```

Now CLEAR the screen and then give the command:

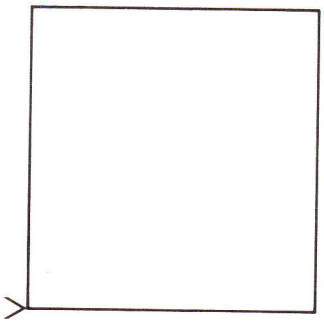
```
VSQUARE
```

A size 20 square will be drawn. Now a new size can be specified before the command VSQUARE, so a square of any size may be drawn.

Try:
```
SIZE 10
VSQUARE
```



Or:
```
SIZE 50
VSQUARE
```

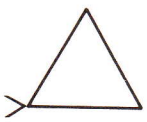


Using three different variables it is possible to define a procedure to draw any regular polygon.

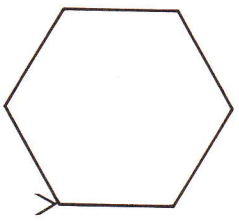
Type:
```
DEFINE POLYGON
REPEAT NUMBER
FORWARD SIZE
LEFT ANGLE
AGAIN
END
```

In response to requests to specify variables, type 4 for NUMBER, 30 for SIZE and 90 for ANGLE. A square will be drawn, and the procedure for a generalised polygon stored. Recall POLYGON by typing:

```
NUMBER 3
SIZE 20
ANGLE 120
POLYGON
```

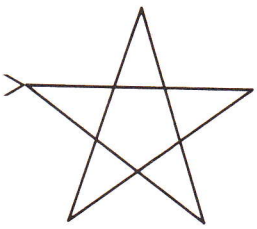


or:
```
NUMBER 6
SIZE 20
ANGLE 60
POLYGON
```



This procedure can also draw stars.

Try:
```
NUMBER 5
SIZE 40
ANGLE 144
POLYGON
```



Further experimentation will be valuable here. In LOGO CHALLENGE circles are in fact many-sided polygons, so this procedure could be used to draw a circle.

> **Note:** More information on drawing circles is given on page 15.

More about EDIT

On page 6, the procedure for editing SQUARE involved altering the size by typing R (Replace) and changing FORWARD 20 to FORWARD 50. Altogether there are three editing commands:

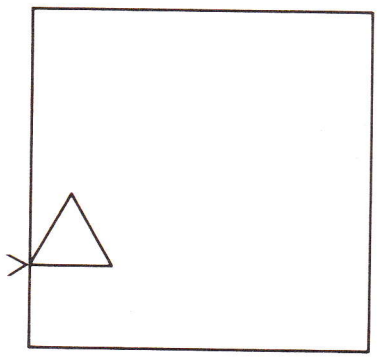
R — to allow a line to be replaced by something else.
D — to delete a line.
I — to allow an extra line or lines to be inserted.

> **Note:** Before proceeding make sure that the procedures for SQUARE, VSQUARE, TRIANGLE and HOUSE are defined and stored in the memory. Type LIST to check this. Edit SQUARE again, changing FORWARD 20 to FORWARD SIZE so that the procedure includes a variable.

Now that SQUARE has been edited to include a variable, that variable must always be specified before using the procedure. If this is not done the procedure will either draw a square of the size which was last specified (possibly some time ago) or, if SIZE has not been specified at all since the program was loaded, the following message "SIZE not set — how big should it be?" will appear.

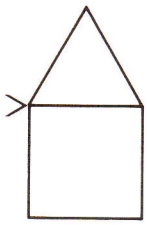
Since the HOUSE procedure incorporates SQUARE, some strange effects could be produced. Try typing:

```
SIZE 80
HOUSE
```



This will give a SIZE 80 square followed by the SIZE 20 triangle positioned as in the original instructions for HOUSE. Clearly, for HOUSE to work, SIZE 20 must be specified.

```
SIZE 20
HOUSE
```



Better still, edit TRIANGLE to include the same variable as SQUARE.

```
EDIT TRIANGLE
RETURN          "REPEAT 3"
RETURN          "FORWARD 20"
R               "Replace:"
FORWARD SIZE
RETURN          "LEFT 120"
RETURN          "AGAIN"
RETURN          "Any More? (Y/N)"
N RETURN        "Command:"
```

Now type:

```
SIZE 10
HOUSE
```



In order to close the gap between the house and roof, HOUSE itself must be edited. A listing of the HOUSE program will show which line is producing the problem. If the line FORWARD 20 is again replaced by FORWARD SIZE the procedure should work. Now try:

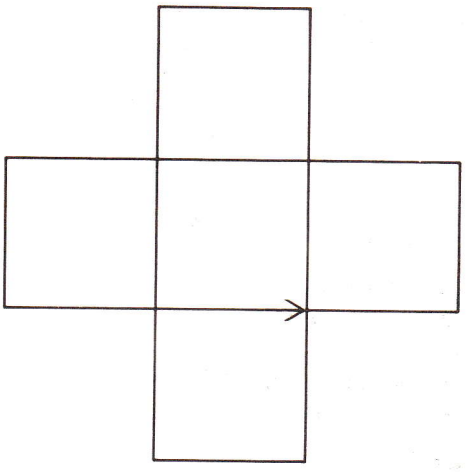
```
SIZE 10
HOUSE
```

or:

```
SIZE 60
HOUSE
```

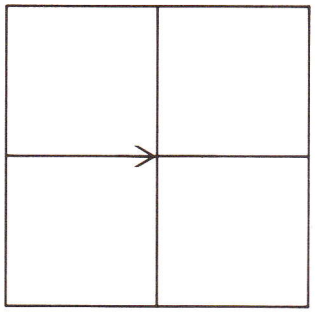
Use of the D (delete) or I (insert) instructions in the EDIT mode is equally simple. D will simply delete the last line and automatically present the next. To try out the use of D, type in the following procedure:

```
DEFINE PLUS
SIZE 30
REPEAT 4
SQUARE
LEFT 90
FORWARD 30
AGAIN
END
```



Now EDIT the procedure PLUS, deleting the line FORWARD 30, as follows:

EDIT PLUS	
RETURN	"SIZE 30"
RETURN	"REPEAT 4"
RETURN	"SQUARE"
RETURN	"LEFT 90"
RETURN	"FORWARD 30"
D	"AGAIN"
RETURN	"Any More? (Y/N)"
N RETURN	"Command:"



The PLUS procedure should now produce a four-pane "window".

Typing I will produce "Insert:", requesting the line or lines to be inserted. Lines can then be typed in, using the RETURN key between each one. When the insertion has been completed use of the RETURN key will restore the EDIT mode and present the next line of the procedure. To practise this, edit PLUS yet again, to put back the FORWARD 30 line, as follows:

EDIT PLUS	
RETURN	"SIZE 30"
RETURN	"REPEAT 4"
RETURN	"SQUARE"
RETURN	"LEFT 90"
RETURN	"AGAIN"
I	"Insert:"
FORWARD 30	
RETURN	"Insert:"
RETURN	"Any More? (Y/N)"
N RETURN	"Command:"

Now clear the screen and type PLUS to check that the picture is restored to its original form. Further experimentation with the use of the edit mode would be valuable. The *Pupil Book* describes the use of "Replace" but not "Delete" or "Insert". Pupils are directed to the teacher for guidance.

Arithmetic with variables

Variables can be used with the four arithmetical functions: addition, subtraction, multiplication (use the * key) and division (use the / key). The value of any of the variables, SIZE, ANGLE, NUMBER and VALUE, may be altered. For example:

```
SIZE = SIZE + 3
SIZE = SIZE / 5
```

Although the values used in commands must be whole numbers, it is possible to multiply by a decimal by combining two instructions. So:

```
SIZE = SIZE *11
SIZE = SIZE / 10
```

will multiply SIZE by 1·1.

Three examples of the use of this feature are given below. The first draws a series of concentric squares, giving a three-dimensional effect which is quite hypnotic.

Note: The first and third programs assume that you have already defined SQUARE as:

```
REPEAT 4
FORWARD SIZE
LEFT 90
AGAIN
```

```
CLEAR
SIZE 20
REPEAT 30
SQUARE
PENUP
BACKWARD 2
RIGHT 90
FORWARD 2
LEFT 90
PENDOWN
SIZE = SIZE + 4
AGAIN
```

The second example draws a spiral.

```
CLEAR
SIZE 2
REPEAT 100
FORWARD SIZE
LEFT 10
SIZE = SIZE *51
SIZE = SIZE / 50
AGAIN
```

To discover what the third example does, type it in.

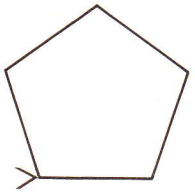
```
CLEAR
PENUP
BACKWARD 90
LEFT 90
BACKWARD 90
RIGHT 90
PENDOWN
SIZE 180
ANGLE 27
ANGLE = ANGLE / 4
REPEAT 24
SQUARE
SIZE = SIZE *17
SIZE = SIZE / 161
FORWARD SIZE
LEFT ANGLE
SIZE = SIZE *145
SIZE = SIZE / 17
AGAIN
```

It is also possible to combine variables in arithmetical statements. For example, a generalised polygon program could be defined as follows:

```
DEFINE POLY
ANGLE = 360 / NUMBER
REPEAT NUMBER
FORWARD SIZE
LEFT ANGLE
AGAIN
END
```

Now only a NUMBER and SIZE need be specified before calling the procedure.

Type:
```
NUMBER 5
SIZE 20
POLY
```



Note: Compare this with the POLYGON procedure defined on page 8.

Section Four

SAVING YOUR WORK

The routines stored in the computer by use of DEFINE can also be stored on a cassette or disk. At the end of a session of working with LOGO CHALLENGE type:

FINISH

The following message will appear on the screen:

"Do you want to save this file? (Y/N)"

If there are some procedures you wish to save stored in the computer's memory, type Y. The following response will appear:

"What shall I call the file?"

Any name of six letters or less will be accepted.

Note: Numbers, spaces or other characters should not be used in file names. An error message will result if an unsuitable name is given or if there is already a file bearing the same name.

After a name has been given, the next step will differ depending on whether a disk or cassette form of storage is being used.

For disk-based machines the procedures are stored on the reverse of the disk in a file bearing the given name.

For cassette-based systems, screen instructions are given for loading a cassette and storing the file. In the *Pupil Book* it is suggested that each pupil has a separate cassette on which to store his/her own file.

Note: When procedures are saved on disk or cassette they will be lost from memory. It may be necessary to experiment first to ensure that procedures are saved and can be reloaded successfully. If problems arise, see Technical Specification on page 25.

To return to the program after storing a file simply type RUN. Now, in response to "Do you have a data file?", type Y. The name will then be requested. Again, with disks loading will be automatic and with cassettes instructions will be given for loading the file.

The file can be updated after each work session. If you have made any alterations to the file, when you type FINISH, the question: "Do you want to save this file? (Y/N)" will appear as usual. However a response of Y will not result in a request for a name because procedures are automatically saved under the same name as before.

If you have made no alterations and defined no new procedures, typing FINISH will just take you out of the program and present "Ready".

On disk, the updated file will be saved on top of the earlier version, which will thus be lost. That is, the new file (which will 'wipe out' the earlier version) will include previously stored procedures that remain unaltered, together with the edited versions of any altered procedures, and completely new procedures.

On cassette, the file may be saved on top of the old one or on a different cassette. If it is not saved directly on top, careful labelling of cassettes will be necessary to avoid confusion between two files which will have the same name.

Having reloaded the file at the start of a session, the command LIST can be used to verify that all the procedures are still intact.

Note: It is a good idea to build up a separate Teacher File containing more complex procedures to offer extra Challenges to the pupils. (See page 23.) The published Answer File should be kept intact for use by others.

Section Five

USING LOGO CHALLENGE WITH CHILDREN

A note about LOGO

In his book *Mindstorms,* Seymour Papert describes how he has used LOGO with children as young as five years old. He also puts forward a stimulating rationale for the use of LOGO with children, claiming that such a tool could revolutionise our whole approach to, and definition of, the educational process.

The essential value of the program lies in its open-endedness and flexibility. Moreover, it is not only child centred but child directed. The pupil is the controller of his/her own learning and intellectual challenges in a way rarely found in the traditional primary school curriculum.

Organisation

LOGO CHALLENGE has been trialled with able children at top junior level (eleven years). However, its potential extends far beyond this, to pupils of all abilities and, possibly, to much younger children. Experience indicates that a working group of three to four children is best because it is small enough to allow all participants to share in hands-on control and decision making. Any less than three would give little scope for discussion and interaction in problem solving. It may also be preferable to have each group fairly well matched in ability. If the range is too great there is a likelihood that the quicker children will solve a problem before the slower ones realise that it exists.

Introducing LOGO CHALLENGE

Every teacher will have different ideas about how to work with LOGO CHALLENGE. The notes which follow are based on experiences of using the program with children, but are not intended as anything more than a general guide. For maximum understanding to develop, however, it is recommended that pupils be introduced to only a few commands at a time.

Initially, children can be introduced to the four commands FORWARD, BACKWARD, RIGHT and LEFT. Each of these commands must, of course, be followed by a number. For FORWARD and BACKWARD the units involved relate to the size of the screen, in terms of graphical characters. There is no need to tell the children this; trial and error is a perfectly adequate way of learning about the length of a unit. Similarly, with younger children, a trial-and-error approach can be used for the amount of turn to left or right. Older children, who will be familiar with angular measurement in degrees, will generally assume that this system is used in LOGO CHALLENGE and suggest that LEFT 90 will produce a 90-degree turn. Children who are not familiar with angular measurement will be inclined to start by typing LEFT 1 and LEFT 5. Such a trial-and-error approach should be encouraged.

Papert describes in his book how a group of youngsters discovered that they could use LEFT 100 followed by RIGHT 10 in a series of instructions to produce a right angle. They used this formula for quite some time before they realised that it could be abbreviated to LEFT 90. Such an individual approach is to be commended, not criticised. The beauty of LOGO is that it allows the children to build their own thinking and reasoning upon the schemata already present. Knowledge is not presented in the form of facts to be learned, but is there for discovery.

Pupils can be challenged to draw a square using FORWARD, BACKWARD, LEFT and RIGHT commands only. They should be allowed to work out their own procedure, but will generally tend to follow that described on page 4. The tedium of typing each instruction four times can be alleviated if the pupils are now introduced to the REPEAT/AGAIN command.

The concept that new commands should be introduced to pupils as and when they are required is central to the LOGO rationale. If REPEAT is introduced because pupils feel a need for it, it will be more thoroughly understood and incorporated into their thinking.

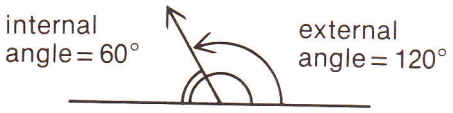
Having drawn a square, pupils can be challenged to draw an equilateral triangle. Some children will happily use the REPEAT function straight away while others will prefer to take things one step at a time. The usual first attempt is:

```
FD 20
LT 60
FD 20
```



This is the standard response from fourth-year junior pupils who know that the internal angles of an

equilateral triangle measure 60 degrees. These commands, however, do not produce quite what was intended. Here is the first bug in the child's series of instructions. Many children are able to solve this problem for themselves. The amount of turn should be given by the *external angle* of the triangle, not the internal. The internal angles of an equilateral triangle are each 60 degrees, thus the amount of turn required will be 180—60 = 120 degrees.



Some junior-school children may not know these terms. One approach to explaining the problem is to get them to pretend that they are the pen and to move in such a way as to draw the triangle on the floor, i.e. to play "pen". This helps them to realise that they are not turning through the internal angle of the triangle but through the remainder of the 180 degrees. This technique of "walking through" problems should be emphasised to pupils. It provides them with a tool for thinking, which is valuable at all stages and may be essential in solving some of the more difficult Challenges.

An alternative approach is to consider the total amount of turn involved in making the full triangle or square. If the pen is to finish with the same orientation and in the same position as it started, it must turn through a full 360 degrees. Thus for a square, with four angles, each turn must be 360/4 = 90 degrees; and for a triangle, with three, it must be 360/3 = 120 degrees.

With younger children the trial-and-error approach may be preferred. However, it would be a pity to waste the opportunity of examining this interesting aspect of mathematics, so trial-and-error solutions should still be discussed and developed.

It should now be possible for pupils to write a program to draw any regular polygon. As before, such activities should be pupil directed, with teacher input only when pupils get totally stuck. For pupils to make mistakes and correct them is the essence of the educational value of the program. Moreover, there is clearly a great deal of satisfaction to be derived from working things out for oneself.

Planning programs

From the beginning pupils should be encouraged to think and plan before they begin typing at the keyboard. Good programs need thinking through carefully and a hasty attack on the keyboard can result in unnecessarily long-winded series of instructions. The *Pupil Book* aims to lead pupils towards what is called, in computer jargon, a "top-down analysis" of the task in hand. This means that they break down the problem into manageable portions or building blocks, a technique essential to success in some of the later Challenges. This is fostered by encouraging planning in advance, on paper. For those pupils for whom this pencil-and-paper stage is so onerous that much of the joy of working with LOGO CHALLENGE is lost, a compromise must be reached. For this reason the *Pupil Book* has a 'softly softly' approach which recommends pencil-and-paper planning but does not insist upon it.

Storing programs

Pupils could now be introduced to the DEFINE mode so that they can store the routines already constructed. The importance of END, to close the definition, should be stressed. Omission of END (or AGAIN in REPEAT mode) produces confusing results.

Combining procedures

In combining the SQUARE and TRIANGLE procedures to form HOUSE, pupils are likely to come across a number of bugs including those described on page 5. Debugging this and other procedures is of immense value. The children will learn that it is not "wrong" to make an error in a program. The appearance of a bug simply offers a challenge to find the bug and correct it. Thus they create their own problems which then require solutions.

Moreover there is usually more than one way to write a working program; there is no one "correct answer" which must be found. No marking is required—if a drawing is completed satisfactorily the problem has been solved and pupils can see their own success instantly. This does not mean, however, that the teacher is redundant. Teacher comments on the strategies and approaches adopted by each child are vital to the development of thinking skills.

PENUP/PENDOWN commands can be introduced as necessary. They are not essential to HOUSE since the pen is moved over an existing line and there is no extra line to mar the diagram.

Introducing variables and editing

At this stage pupils may be ready for introduction to the variables SIZE, ANGLE, NUMBER. For example, having defined HOUSE, one group of pupils suggested drawing a number of houses to form a street. However HOUSE had been defined using a large FORWARD instruction (FORWARD 80) and so the house was too large to fit on the screen more than twice. They asked if there was any way the house could be made smaller without rewriting SQUARE, TRIANGLE and HOUSE. Here was an obvious point at which to introduce both the

variables and the EDIT mode. Other pupils may wish to spend several sessions familiarising themselves with the earlier work before they are ready for variables or editing.

Ideally, as stressed above, the introduction of new commands should be initiated by a query from pupils rather than by teacher direction.

Drawing circles

A group of pupils left to "play" with LOGO CHALLENGE frequently opts to draw people or vehicles. In either case a circle is called for. So far pupils have only been manipulating straight lines. Some children may duck out of the problem and make do with square heads or wheels, and in such cases the teacher may wish to input a challenge: "How can you draw a circle?".

One approach is to get the children to play "pen"—to walk in a circle. When asked to analyse their movements, one group of pupils responded: "Well you go forward a bit and turn a bit." This idea was developed by asking the children to consider the size of "a bit" and the total amount of turn. The group (who were fourth-year pupils) thought that 360 should figure somewhere, and produced the following procedure:

```
DF CIRCLE
RP 360*
FD 1
LT 1
AGAIN
END
```

This indeed produced a large circle, but it was clearly too big to be a head or a wheel in a larger drawing. How, they wondered, could they draw a smaller circle? Again the pupils played "pen" and decided that it would be necessary to "turn more".

> ***Note:** Normally numeric commands must not use numbers larger than 180. The only exception is 360, which will be accepted because it is so obviously useful.

Next, the group tried:

```
RP 360
FD 1
LT 5
AGAIN
```

This certainly drew a smaller circle, but it took a great deal of time as the cursor circumnavigated the circle five times. The pupils quickly spotted that the bug was in the line RP 360. As each turn was now 5 degrees, only 360/5 = 72 repeats were necessary.

Lastly, by replacing FD 1 by FD SIZE, circles of variable size could be drawn.

The generalised polygon program on page 8 may be used to draw circles of any size. This illustrates the point that in LOGO CHALLENGE circles are really many-sided polygons. Pupils may wish to experiment to discover how few sides are necessary to produce something that looks like a fairly smooth circle.

> **Note:** Circle drawing is much faster with the arrowhead hidden.

From here onward the children are limited only by their own imagination. They are able to work with the program with very little teacher guidance, creating and storing images and setting up and solving problems.

Variables with arithmetic

When pupils are ready, introduce the use of arithmetical functions in combination with variables, to draw spirals for example. Quarter-circles can be combined to form leaves or petals; petals combined to form flowers . . . the potential is limitless.

Again, pupils will differ in the amount of guidance that they require. Free exploration may be engrossing to one child while another will quickly become bored with it. The *Pupil Book* aims to maintain the interest of the second child by offering a series of graded Challenges.

The challenge of LOGO CHALLENGE

Using LOGO CHALLENGE with children should prove an exciting and stimulating experience. It may, however, be necessary to learn a new humility. In computer-programming courses at secondary level, pupils frequently progress beyond their teachers. The use of LOGO CHALLENGE may well place primary teachers in a similar position.

How well will this challenge be met?

Section Six

NOTES ON CHALLENGES IN THE PUPIL BOOK

Challenge 1

As explained on page 7, the dimensions of the screen are:

X from −160 to +160
Y from −90 to +90

The screen is thus 320 units wide. However, if pupils endeavour to keep the arrowhead on the screen at the left-hand side they will come up with an answer somewhat less than 320. They may also discover that the program will not accept numbers larger than 180 (except for 360—see page 15), and so they will have to move the pen in several steps.

Challenge 2

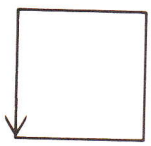
If the pupils are familiar with the concept of a right angle, they will probably assume that measurement is in degrees of angle. If they do not assume this, it is better to allow them to experiment than to tell them that 90 is the number to use (see page 13). Any length of line could be used here, and there is no need for the two lines to be the same length.

Challenge 3

Again, any length of side could be chosen. The square may be drawn with LEFT or RIGHT turns, or even by moving backwards. Original approaches should be encouraged.

Sample solution:

```
FORWARD 50
LEFT 90
FORWARD 50
LEFT 90
FORWARD 50
LEFT 90
FORWARD 50
```

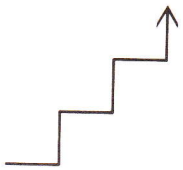


Challenge 4

There is no need for pupils to match their constructions precisely to the pictures in the book. They may, for example, prefer to draw more steps or to start with an upstroke.

Sample solution:

```
FORWARD 20
LEFT 90
FORWARD 20
RIGHT 90
FORWARD 20
LEFT 90
FORWARD 20
RIGHT 90
FORWARD 20
LEFT 90
FORWARD 20
```

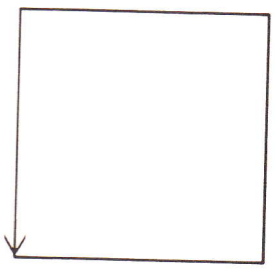


Challenge 5

The purpose of this exercise is to practise moving the pen before starting the drawing.

Sample solution:

```
PENUP
BACKWARD 50
RIGHT 90
FORWARD 50
LEFT 90
PENDOWN
FORWARD 100
LEFT 90
FORWARD 100
LEFT 90
FORWARD 100
LEFT 90
FORWARD 100
```



The instructions bracketed together move the pen to the start position without leaving a trace.

Challenge 6

As suggested in the *Pupil Book,* use of a pencil and paper to design the letters is a good idea. Pupils may need to move the pen before starting the first letter, and it will certainly be necessary to move it between letters. PENUP and PENDOWN commands can be used for this. The series of instructions required to complete this Challenge will probably be quite long. UNDO can be used to correct mistakes.

Challenge 7

Again, a lengthy series of instructions will be

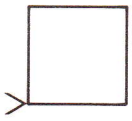
needed to construct any of the drawings suggested. Pupils may prefer to work on their own ideas. If they do so, pencil-and-paper planning should be encouraged. The purpose of the exercise is to practise using the keyboard and the LOGO CHALLENGE commands encountered so far. Later, pupils will be introduced to the use of REPEAT and to procedures which will enable them to produce these pictures more efficiently.

Challenge 8

In order to produce a square, something will have to be repeated four times. Only two lines will be needed inside the REPEAT/AGAIN loop. When the square is complete the arrowhead will be reset to its starting position (facing East), which was not necessarily the case when the square was drawn without using REPEAT. It is worthwhile asking pupils to think about why this is so. They should also be encouraged to think about what changes and what stays the same when the size of the square is altered. This is a simple approach to the concept of congruence.

Sample solution:

```
REPEAT 4
FORWARD 40
LEFT 90
AGAIN
```

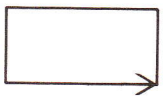


Challenge 9

For the rectangle, the repeated unit comprises two sides not one. Thus there will be four lines inside the REPEAT loop and only two REPEATS will be needed. Analysis of the differences between a square and a rectangle should be encouraged. As with all the Challenges, a large number of alternative solutions are possible.

Sample solution:

```
REPEAT 2
LEFT 90
FORWARD 30
LEFT 90
FORWARD 60
AGAIN
```

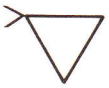


Challenge 10

The main difficulty with this Challenge is likely to be finding the angle of turn. Page 14 describes how this could be approached. "Walking through" the problem is a good strategy and can be used on many occasions. Much of the value of LOGO CHALLENGE will be lost if pupils are simply told that the number to use is 120; it is essential that they make the discovery for themselves.

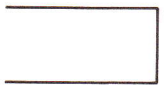
Sample solution:

```
REPEAT 3
FORWARD 30
RIGHT 120
AGAIN
```

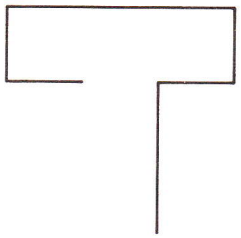


Extra Challenges using REPEAT/AGAIN loops

The purpose of these activities is for pupils to practise using REPEAT loops and discover how such loops can considerably shorten a set of instructions. Planning is important, and pencil-and-paper work away from the computer may also help to ameliorate problems of congestion at the keyboard. The *Think!* clouds contain suggestions which should help at the planning stage. The first step is always to identify the principal unit or building block, and to see how many times it must be repeated. This can be done visually before any encoding is attempted. For PLUS, this unit must be repeated four times.

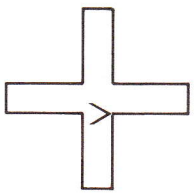


For CORRIDORS, this unit must be repeated twice.



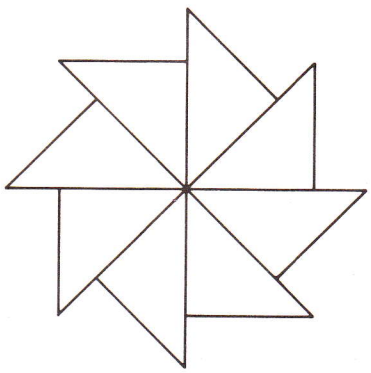
Sample solution for PLUS:

```
REPEAT 4
FORWARD 30
LEFT 90
FORWARD 10
LEFT 90
FORWARD 30
RIGHT 90
AGAIN
```



Sample solution for WINDMILL:

```
REPEAT 8
FORWARD 50
LEFT 90
FORWARD 50
LEFT 135
FORWARD 71
LEFT 180
AGAIN
```



In drawing WINDMILL pupils must think carefully about the angles through which to turn, and the length of the hypotenuse of the triangle which comprises the building block. Trial-and-error solutions are possible, although slight distortions in the drawing may arise. Investigations of Pythagoras' theorem and the use of electronic calculators to find square roots could arise as extensions of the work. Remember, the program will not accept fractions or decimals so square roots will have to be taken to the nearest whole number.

In order to achieve the symmetry of MOUNTAINS and BATTLEMENTS extra commands must be added outside the REPEAT loop.

Sample solution for MOUNTAINS:

```
REPEAT 3
LEFT 80
FORWARD 30
RIGHT 80
FORWARD 15
RIGHT 80
FORWARD 30
LEFT 160
FORWARD 30
RIGHT 160
FORWARD 30
LEFT 80
AGAIN
LEFT 80
FORWARD 30
RIGHT 80
FORWARD 15
RIGHT 80
FORWARD 30
```

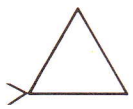


Challenge 11

Pupils are asked to define a procedure for a triangle with the same length of side as the square defined in Lesson E. This is in preparation for Challenge 14, where they will be challenged to put the two shapes together to form a house. Thus the side length is specified and should be adhered to.

Sample solution:

```
DEFINE TRIANGLE
REPEAT 3
FORWARD 40
LEFT 120
AGAIN
END
```



Challenge 12

The staircase which was drawn in Lesson D could be used again here. The set of instructions should begin: DEFINE STAIRCASE and conclude with: END.

Pupils may prefer to store a staircase which starts at the bottom of the screen, in which case the preliminary pen movements could also be included in the definition. Such a staircase would, however, be less versatile.

Challenge 13

This is an open-ended exercise designed to give practice in storing procedures.

Challenge 14

The process of combining SQUARE and TRIANGLE to form HOUSE is described on pages 5 and 9. This Challenge should present many pupils with ample opportunities for debugging. However, if these procedures have already been defined appropriately, the instructions:

```
SQUARE
TRIANGLE
```

will give a house, in which case the question, "Why does this not give a house?" will be redundant.

Challenge 15

DEFINE HOUSE . . . END is used to store the procedure. Pupils are referred to earlier lessons for information on updating their disk or cassette file. Further details are also given on page 12 of this *Guide.*

Challenge 16

One approach to drawing a street is given on page 6. Terraced houses can be drawn with a shorter set of instructions.

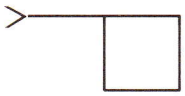
Extra Challenges using procedures

Using procedures to build up programs is an important concept in computer programming. It is one element of an approach generally known as "top-down analysis", which is a useful strategy for tackling problems. In top-down analysis the whole problem is broken down into its component parts, each of which is attacked separately. Applied to LOGO CHALLENGE, this means breaking down a picture or pattern into discrete building blocks, defining a procedure for one block, and then putting the blocks together to assemble the whole picture.

It should be noted that there is still a multitude of solutions to any problem, even when this technique is used. For example, the building block used in the *Pupil Book* for PLUS is not the same as that suggested on page 17 of this *Guide.*

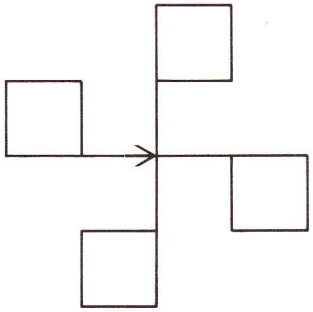
Sample solution for FLAGS:

Stage 1



```
DEFINE FLAG
FORWARD 60
REPEAT 3
RIGHT 90
FORWARD 30
AGAIN
RIGHT 90
BACKWARD 30
END
```

Stage 2



```
DEFINE FLAGS
REPEAT 4
FLAG
LEFT 90
AGAIN
END
```

Sometimes a number of different building blocks may be defined, as in this method of constructing BATTLEMENTS.

Sample solution for BATTLEMENTS:

Stage 1



```
DEFINE TALL
LEFT 90
FORWARD 40
RIGHT 90
FORWARD 20
RIGHT 90
FORWARD 40
END
```

Stage 2



```
DEFINE SMALL
LEFT 90
FORWARD 20
RIGHT 90
FORWARD 10
RIGHT 90
FORWARD 20
END
```

Stage 3



```
DEFINE GAP
LEFT 90
FORWARD 10
END
```

Stage 4

```
DEFINE BATTLEMENTS
REPEAT 3
TALL
GAP
SMALL
GAP
SMALL
GAP
AGAIN
TALL
END
```



Challenge 17

There is little new thinking involved in this Challenge, the purpose of which is mainly to practise the use of EDIT. Page 9 gives more guidance on editing.

Challenge 18

This interesting exercise in debugging is explained in detail on page 9. The Challenge also introduces pupils to listing procedures, which are important steps in debugging. LIST MARMALADE will produce an error message, as no procedure called MARMALADE has been defined.

Challenge 19

This exercise gives further practice with EDIT. The *Pupil Book* does not contain any information about using Insert or Delete in EDIT mode, instead pupils are referred to the teacher for guidance. Page 9 of this *Guide* covers editing in some detail and gives examples of its use. Further notes on the HELP file and WHERE appear on page 7.

Challenge 20

It will probably be necessary to edit SQUARE, TRIANGLE and then HOUSE itself, depending on how HOUSE was defined. Pupils should use LIST HOUSE to check on the instructions used (see page 6).

Challenge 21

This picture differs from that produced in Lesson H in that the squares do not all have a common starting point. Pupils should think carefully about what extra instructions will be required. These could be defined as a separate procedure.

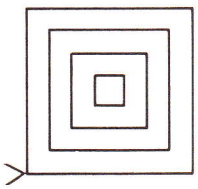
Sample solution:

Stage 1



```
DEFINE MOVE
PENUP
BACKWARD 10
RIGHT 90
FORWARD 10
LEFT 90
PENDOWN
END
```

Stage 2



```
DEFINE WELL
SIZE 10
SQUARE
MOVE
SIZE 30
SQUARE
MOVE
SIZE 50
SQUARE
MOVE
SIZE 70
SQUARE
MOVE
END
```

Challenge 22

There is no new thinking involved here. The purpose of the exercise is to give further practice in the use of variables.

Extra Challenges - regular polygons and circles

These Challenges constitute a distinct unit, which could be used earlier as they do not have to involve use of EDIT or variables. "Walking through" the problems is a useful strategy (see pages 14 and 15).

Challenge 23

This Challenge encourages top-down analysis. FLOWER is constructed from PETAL, and PETAL can be formed using the ARC procedure defined in Lesson I (as REPEAT NUMBER, FORWARD 1, LEFT ANGLE, AGAIN).

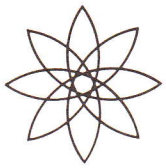
Sample solution:

Stage 1



```
DEFINE PETAL
NUMBER 30
ANGLE 2
REPEAT 2
ARC
LEFT 120
AGAIN
END
```

Stage 2



```
DEFINE FLOWER
REPEAT 8
PETAL
LEFT 45
AGAIN
END
```

Challenge 24

Pupils are asked to define the essential elements of a polygon. To draw any regular polygon a forward movement followed by a turn must be repeated a number of times.

Sample solution:

```
DEFINE POLYGON
REPEAT NUMBER
FORWARD SIZE
LEFT ANGLE
AGAIN
END
```

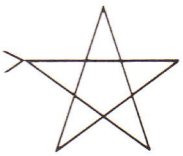
NUMBER 5, SIZE 20, ANGLE 72, POLYGON will produce a pentagon. For a hexagon the settings will be NUMBER 6, SIZE 20, ANGLE 60. Pupils could be encouraged to notice the link between the ANGLE and NUMBER settings (ANGLE will always be 360/NUMBER). Later they will be able to use this arithmetic in the definition.

Challenge 25

The polygon procedure can also be used to draw stars, when NUMBER and ANGLE are not arithmetically linked in the procedure. Star drawing can be very interesting. Is it possible to draw stars with any number of points, using this technique? Opportunities for extensive investigation can be found.

Sample solution:

```
NUMBER 5
SIZE 60
ANGLE 144
POLYGON
```



Pupils should be encouraged to think about why the angle should be 144. Again, "walking through" the problem will help. What is the total amount of turn required to draw this star? How is this pentagram related to a pentagon?

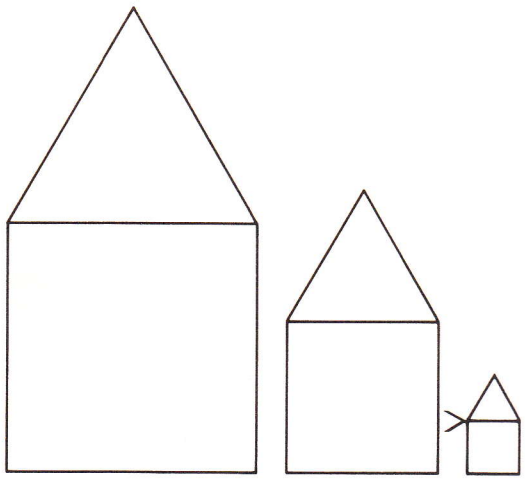
Challenge 26

Provided that HOUSE has been defined to include variables, this Challenge should present no problems. The pen must be moved before starting.

By this stage, pupils should be in the habit of storing all pictures by using DEFINE/END.

Sample solution:

```
DEFINE PERSP
PENUP
BACKWARD 120
RIGHT 90
FORWARD 60
LEFT 90
PENDOWN
SIZE 100
REPEAT 3
HOUSE
FORWARD SIZE
RIGHT 90
FORWARD SIZE
LEFT 90
PENUP
FORWARD 10
PENDOWN
SIZE = SIZE -40
AGAIN
END
```



Challenge 27

A variable circle will be required. This could be defined as follows:

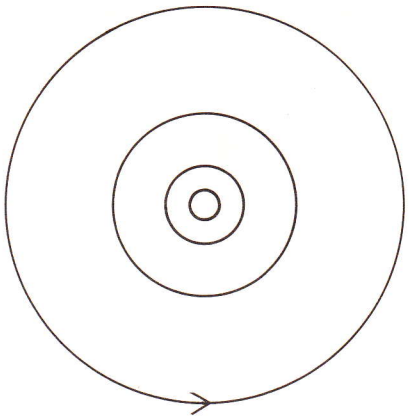
```
DEFINE VCIRCLE
REPEAT 36
FORWARD SIZE
LEFT 10
AGAIN
END
```

By varying SIZE, circles of different sizes can be produced. SIZE 2, VCIRCLE will produce a very small circle; SIZE 10, VCIRCLE will produce a very large one.

This is a difficult Challenge and pupils will have to think carefully about positioning the pen before the start of each new circle.

Sample solution:

```
DEFINE CONC
SIZE = 2
REPEAT 4
VCIRCLE
PENUP
RIGHT 90
FORWARD SIZE
FORWARD SIZE
FORWARD SIZE
FORWARD SIZE
FORWARD SIZE
LEFT 90
PENDOWN
SIZE = SIZE *2
AGAIN
END
```

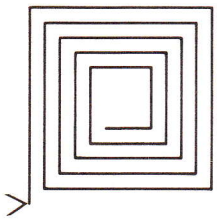


Challenge 28

Spiral drawing can be the basis of an extensive piece of investigatory work. A number of interesting examples are included on the Answer File. Probably the easiest spiral to begin with is one that uses right-angle turns, because the length of the lines is incremented while the angle remains constant.

Sample solution:

```
DEFINE SP
SIZE 10
REPEAT 20
FORWARD SIZE
LEFT 90
SIZE = SIZE + 2
AGAIN
END
```



Other kinds of spiral may involve incrementing the ANGLE only, or both ANGLE and SIZE. Refer to the Answer File for further details.

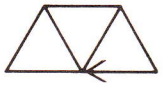
Extra Challenges - elegant programming

Here the Challenge is to use procedures and REPEAT to write very concise programs. The first five are all built from TRIANGLE or from each other. If TRIANGLE includes a variable it is assumed that this will be specified first, outside the definition.

> **Note:** All the following procedures assume that TRIANGLE has been defined as:
>
> ```
> REPEAT 3
> FORWARD SIZE
> LEFT 120
> AGAIN
> ```

Sample solution for TRAPEZIUM:

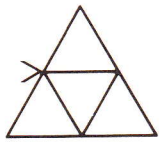
```
REPEAT 3
TRIANGLE
LEFT 60
AGAIN
```



If the TRAPEZIUM procedure is now defined it can be used as a building block for the next Challenge.

Sample solution for BIGTRI:

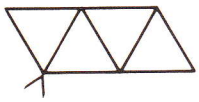
```
TRAPEZIUM
RIGHT 60
FORWARD SIZE
RIGHT 120
TRIANGLE
```



PARALLELOGRAM can also be built using TRAPEZIUM.

Sample solution for PARALLELOGRAM:

```
TRAPEZIUM
FORWARD SIZE
RIGHT 120
TRIANGLE
```

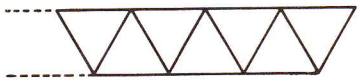


In a similar way, TESSELLATION can be built up from PARALLELOGRAM, using two other procedures—START and MOVE.

Sample solution for TESSELLATION:

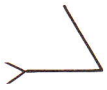
Stage 1

```
DEFINE ROW
REPEAT 6
PARALLELOGRAM
RIGHT 60
BACKWARD SIZE
AGAIN
SIZE = SIZE *13
FORWARD SIZE
SIZE = SIZE / 13
END
```



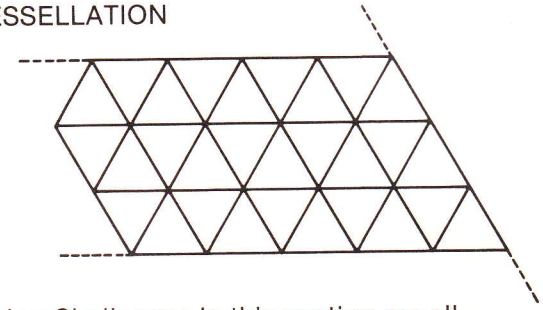
Stage 2

```
DEFINE MOVE
RIGHT 60
FORWARD SIZE
LEFT 60
BACKWARD SIZE
END
```



Stage 3

```
DEFINE TESSELLATION
SETX 130
SETY 75
REPEAT 8
ROW
MOVE
AGAIN
END
```



The remaining Challenges in this section are all part of the same family of patterns, built up by rotating a single-unit polygon. A generalised solution would be:

```
REPEAT ?
?
LEFT ?
AGAIN
```

Careful analysis of the pictures should enable pupils to work out what should take the place of each question mark.

Sample solution for MOSAIC:

```
REPEAT 8
OCTAGON
LEFT 45
AGAIN
```

This assumes that a procedure for OCTAGON has already been defined.

Section Seven

THE ANSWER FILE

Loading

An Answer File called TEACH is provided with the program, on the reverse side of the cassette or disk. It can be loaded as follows.

After loading the program, the LOGO CHALLENGE title page is displayed. After pressing RETURN you will be asked whether you have a data file that you wish to load. Type Y for yes. When asked for the name of the file, type TEACH. The Answer File will then be loaded automatically from disk, or, in the case of a cassette-based system, instructions for loading will be given on the screen.

Note: To consult the TEACH file during the course of a LOGO CHALLENGE session, type FINISH in the usual way and store any defined procedures that you wish to save. You can then type RUN and load the TEACH file as described above (see also page 12).

Uses

The Answer File can be used in a variety of ways:

1 To clarify problems while working through Sections Two and Three of the *Guide.* All the procedures illustrated, such as STREET, are already defined on the TEACH file and can be listed as required. Thus the file can serve as an "answer book" for teacher reference.

2 For class or group discussions of certain procedures or families of procedures. For example, when analysing STREET, the procedures for SQUARE, TRIANGLE, HOUSE and STREET could be displayed in turn.

3 To present additional Challenges to pupils. The TEACH file contains a number of extra procedures for drawings such as spirals, which could be demonstrated to pupils as examples of what it is possible to achieve. Provided pupils are not shown the listings of the procedures, they can be challenged to create similar drawings for themselves.

Note: Once the Answer File, TEACH, is loaded there will be no memory space for defining and storing extra commands. (See Technical Specification on page 25.)

Storing extra Challenges

As mentioned on page 12, we recommend that you build up a separate Teacher File of supplementary Challenges of your own devising. Many ideas could be developed and stored during a planning session, and then be demonstrated and presented to pupils as Challenges, in the same way as the additional spirals. A number of suggestions for your Teacher File are given below.

1 Draw a series of concentric triangles or hexagons.

2 Draw scalene and isosceles triangles.

3 Define procedures for the letters N, S, E and W. Draw a picture of a compass with these points marked. Then put in NE, NW, SE and SW; then NNE, ENE, and so on until there is no more room.

4 Experiment further with spirals. Define a generalised spiral procedure containing four variables. Holding three of them constant, vary just one at a time. What does each instruction control?

5 Try this:

```
REPEAT 8
FORWARD 60
LEFT 45
FORWARD 90
LEFT 90
AGAIN
```

Can more shapes of the same family be constructed?

To save your Teacher File use the usual method, which is explained in detail on page 12.

Glossary of LOGO CHALLENGE Commands

FORWARD (FD)
BACKWARD (BD)
CLEAR (CL)
HOME (HM)
UNDO
LEFT (LT)
RIGHT (RT)
REPEAT (RP) AGAIN (AG)
DEFINE (DF) END (EN)
PENUP (PU)
PENDOWN (PD)

DESCRIBE (DS)
LIST (LS)
DELETE (DL)
EDIT (ED) R (Replace) D (Delete) I (Insert)
NUMBER/SIZE/ANGLE/VALUE (Variables)
FINISH
SETX (SX) SETY (SY)
SCALE (SC)
HELP (HP)
HIDE (HD) SHOW (SH)
WHERE (WH)

Technical Specification

Although the various versions of LOGO CHALLENGE are outwardly very similar, there are some differences.

1 Due to the way the machines handle high-resolution graphics, HELP and LIST erase the picture being displayed on the ZX SPECTRUM and BBC Acorn machines, but not on the RML. Hence HELP and LIST may be called from within DEFINE and REPEAT commands *only in the RML versions.*

2 There is a short 'beep' to signal that the program is waiting for the next command—this is not implemented on the RML 380Z, as there is no internal speaker.

3 The cassette versions of the program issue instructions for the loading and saving of data files; these are not necessary for the disk version and are not included.

4 If the saving routine fails for some reason (e.g. forgetting to press RECORD) it is possible to return to the program by typing:

GOTO 80 (SPECTRUM)
GOTO 3 (RML 380Z)
GOTO 10 (RML 480Z)
GOTO 225 (BBC)

This will return you to the "Command:" prompt. Typing CLEAR will clear the screen of any unwanted messages and allow you to continue.

5 The maximum length for a *file name* varies from system to system—the simplest answer is to limit file names to a maximum of 6 letters. *Command names* may be any length, although 10 characters should be enough!

6 The SPECTRUM saves its data files in a different way to the other machines. This results in several extra messages occurring when files are saved or loaded—these are normal; the user should merely press ENTER when requested.

7 It is important that the [BREAK] key is not pressed on the BBC Acorn machine as the program and any stored procedures will be lost. To continue, the program must be loaded again.

8 If, for any reason, you wish to erase an individual data file from the RML disk, type:

ERA C: file name. LG1

If required, *all* the files on the disk can be erased by typing: ERA C: *. LG1 (if in CPM)
Please note that this will also erase the TEACH *file.*

9 If your BBC Acorn Model B machine has been upgraded with a disk interface, it will be necessary to type the following loading instructions:

```
* TAPE
PAGE = &EØØ
CHAIN "CLOGO"
```

10 If your BBC Acorn Model B machine has been upgraded with an Econet interface, follow the loading instructions as in (9) above.

11 The program can accept up to 85 commands, of which 25 are already part of the program. If you define long commands you may run out of memory. If the Answer File, TEACH, is loaded there will be no memory for defining and storing extra commands.

12 Loading instructions:

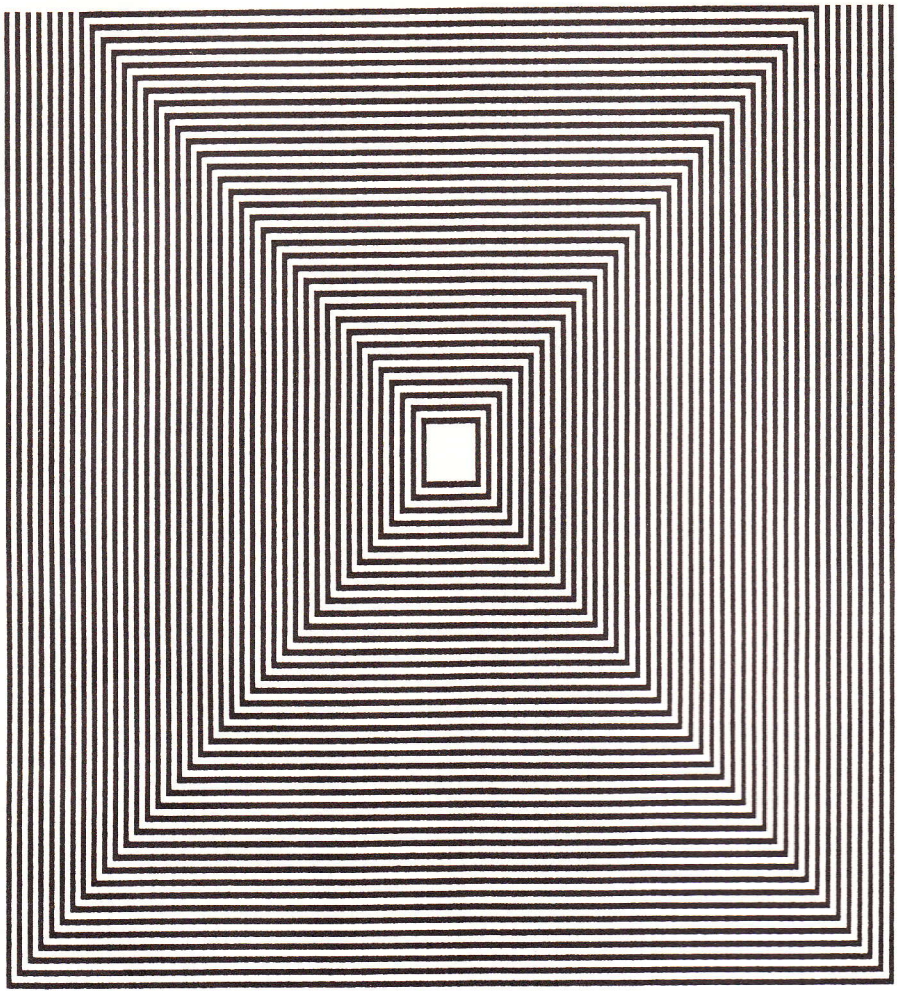
BBC Model B cassette
CHAIN "CLOGO"

RML 480Z cassette
LOADGO "CLOGO"

RML 380Z disk
Press reset button then type B

Sinclair ZX Spectrum cassette
LOAD "CLOGO"

13 DO NOT FORGET TO CONSULT THE *USER GUIDE* FOR YOUR MICROCOMPUTER. Most day-to-day queries can be solved by reference to a *User Guide.*



Logo Challenge is an interactive problem-solving program which simulates the 'turtle graphics' of the LOGO language.

The program and accompanying teaching materials have been developed with children aged 9 – 13 years but are equally appropriate for those younger or older.

The program involves children in programming a microcomputer to draw an infinite variety of pictures and patterns. In response to an appropriate sequence of commands, a 'screen pen' can be made to draw geometric shapes and designs.

Any drawn picture can be stored and used in more complex drawings. The stored procedures can be saved on an individual or group 'file' for use at a later date.

Logo Challenge is most effective with groups of 3 to 5 children to allow participation in decision making. It can be used time and time again without repetition and with a varied age and ability range.

The Pupil Book provides a series of structured *Lessons* on using the program commands. Following each *Lesson* there are *Challenges* enabling children to explore and experiment with the 'turtle graphic' features of the program.

The Teacher's Guide adopts a teach yourself approach to all the commands and includes notes on likely outcomes to many of the *Challenges* in the Pupil Book.

The program also includes an Answer File, for use by the teacher, which illustrates all the *Challenges* in the Pupil Book.

Logo Challenge is available as a Teaching Pack in four different versions for the following machines:
Cassette versions for BBC Acorn Model B, Research Machines 480Z and Sinclair ZX Spectrum.
Disk version for Research Machines 380Z.

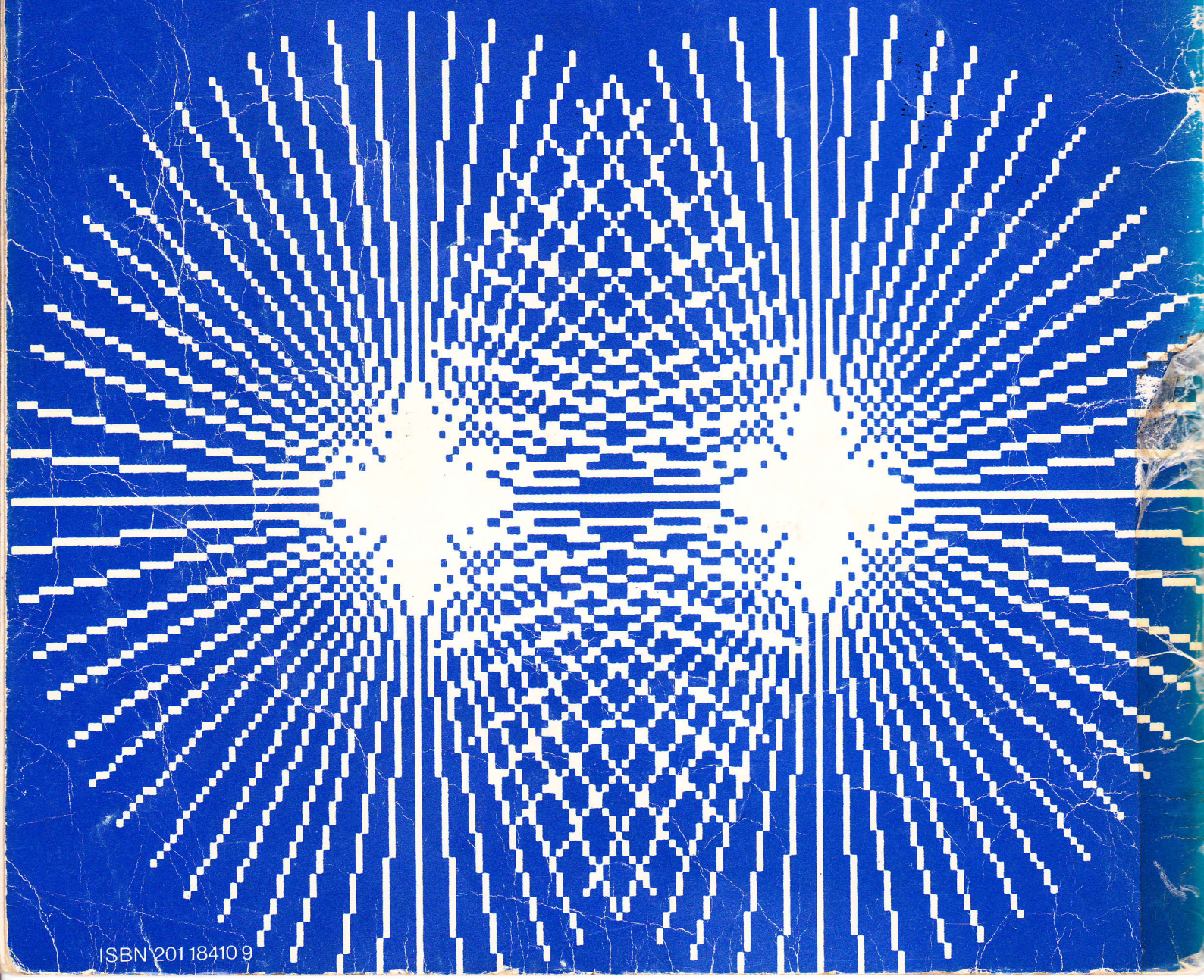
Each Teaching Pack contains:

- One cassette or disk containing the program and answer file.
- One back-up cassette or disk.
- One copy of the Pupil Book.
- One copy of the Teacher's Guide.

The Pupil Book is also available separately in packs of five.



Addison-Wesley Publishers



ISBN 201 18410 9