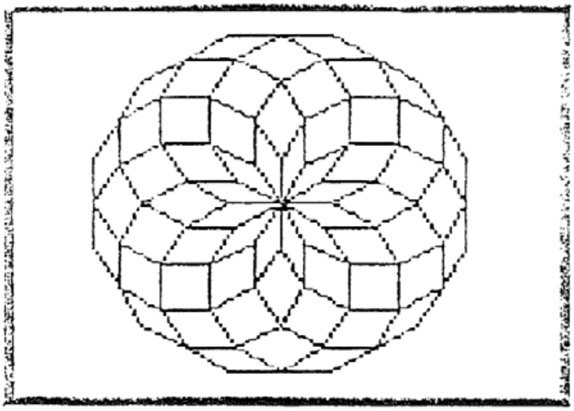
LOGO

some program ideas

by C M Robinson



PATTERN 12

```
TO PATTERN :sides
HT
REPEAT :sides [F
    [FD 800 / :sic
     T 360 / :side
    :T 360 / :side
    ]
```

Avon

Information Technology Centre

About this booklet

I have listed a number of simple LOGO programs in this booklet for you to type in and try. (The usual way to get them running is to type GO followed by RETURN.)

I have also suggested ideas for altering the programs or making them better.

But please remember: These programs are only suggestions, to help you start writing LOGO programs of your own.

A LOGO BIRTHDAY CARD

```
TO GO
  PRINT [What's the name of today's
    lucky boy or girl?]
  MAKE "name RL
  MAKE "counter 1
  BIRTHDAY
END

TO BIRTHDAY
  TYPE [Happy birthday\ ]
  IF :counter = 3 [PRINT SE "dear
    :name] [PRINT [to you]]
  IF :counter = 4 [STOP]
  MAKE "counter :counter + 1
  BIRTHDAY
END
```

What does the 'backslash' (\ ) do?
And what is the difference between PRINT and TYPE?

While using these programs, you will learn new PRIMITIVES. These are the words LOGO understands all the time. The words you teach LOGO are called PROCEDURES. I have always put primitives and procedures in capital letters.

VARIABLES are the names of the 'boxes' where we store information which can VARY. I have used small letters for variables.

If you want to know what a primitive does, why not EXPERIMENT by changing the programs?

QUIZ

```
TO GO
  MAKE "questions [[What is the capital
    of France?] [Who invented LOGO?]
    [4 x 8]]
  MAKE "answers [[Paris] [Seymour
    Papert] [32]]
  QUIZ
END

TO QUIZ
  MAKE "number 1 + RANDOM COUNT :questions
  PRINT CHOOSE_Q
  IF RL = CHOOSE_A [PRINT [That's right!
    ] SOUND 1 -15 116 5 SOUND 1 -15 100 5]
    [PRINT SE [No! It's] CHOOSE_A SOUND
    1 -15 20 10]
END

TO CHOOSE_Q
  OP ITEM :number :questions
END

TO CHOOSE_A
  OP ITEM :number :answers
END
```

Now you could write your own questions and answers.

WRITING SENTENCES

```
TO GO
  MAKE "nouns [Jack Jill Mary John
    Matilda]
  MAKE "adjectives [Ugly Beautiful
    Short Tall Fat Thin Jolly]
  MAKE "verbs [stutters speaks talks
    thinks]
  MAKE "adverbs [slowly sometimes
    quickly thoughtfully carefully]
  TS
  REPEAT 5 [PHRASES]
END


TO PHRASES
  MAKE "noun ITEM 1 + RANDOM
    ( COUNT :nouns ) :nouns
  MAKE "adjective ITEM 1 + RANDOM
    ( COUNT :adjectives ) :adjectives
  MAKE "verb ITEM 1 + RANDOM
    ( COUNT :verbs ) :verbs
  MAKE "adverb ITEM 1 + RANDOM
    ( COUNT :adverbs ) :adverbs
  PRINT ( SE :adjective :noun
    :verb :adverb ". )
END
```

Try altering this program using different words, and making different types of sentences. Perhaps you could make it write sentences similar to:
"The cat chases the frightened mouse."
(You could use sets of words called "subject and "object.)

TUNE (Making music)

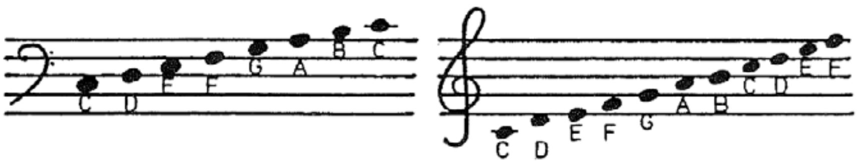
```
TO GO
  MAKE "channel 1
  REPEAT 2 [PHRASE1]
  PHRASE2 LINK1 PHRASE2
  REPEAT 2 [PHRASE3 LINK2]
  PHRASE3 LINK3
  PHRASE1
END

TO PHRASE1
  MAKE "notes [77 69 61]
  MAKE "length [12 12 24]
  MAKE "volume [-15 -15 -15]
  PLAY
END

TO PHRASE2
  MAKE "notes [89 81 81 81 77]
  MAKE "length [12 7 1 4 20]
  MAKE "volume [-15 -15 0 -15 -15]
  PLAY
END

TO LINK1
  MAKE "notes [77]
  MAKE "length [4]
  MAKE "volume [-15]
  PLAY
END

TO PHRASE3
  MAKE "notes [89 109 109 109 105 97 105]
  MAKE "length [4 7 1 4 4 4 4]
  MAKE "volume [-15 -15 0 -15 -15 -15 -15]
  PLAY
END

TO LINK2
  MAKE "notes [109 89 89 89 89]
  MAKE "length [8 3 1 7 1]
  MAKE "volume [-15 -15 0 -15 0]
  PLAY
END
```

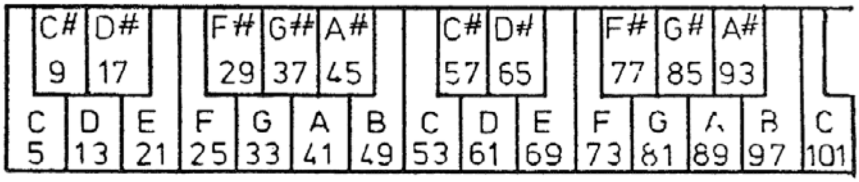
```
TO LINK3
  MAKE "notes [109 89 89 89 89 89 81]
  MAKE "length [4 3 1 3 1 8 4]
  MAKE "volume [-15 -15 0 -15 0 -15 -15]
  PLAY
END
```

```
TO PLAY
  MAKE "l FIRST :length
  MAKE "length BF :length
  MAKE "n FIRST :notes
  MAKE "notes BF :notes
  MAKE "v FIRST :volume
  MAKE "volume BF :volume
  SOUND :channel :v :n :l
  IF EMPTY? :length [STOP]
  PLAY
END
```

To find out what tune this plays, you will have to try it out.
How can it be made to play faster?

This stave and keyboard (with note values) will help you to write your own tunes.





A SIMPLE DATABASE

```
TO GO
  START
  INPUT
  SEARCH_FOR
  DISPLAY
END

TO START
  MAKE "name [John Jack Mary]
  MAKE "house_no [21 32 43]
  MAKE "road [[Longan Winding Way]
    [Micro Drive] [Finished Avenue]]
  MAKE "hobby [Football [Stamp
    collecting] Swimming]
  MAKE "counter 1
END

TO INPUT
  TS
  PRINT [Press 1 to search names] PRINT []
  PRINT [Press 2 to search roads] PRINT []
  PRINT [Press 3 to search hobbies] PRINT []
  INKEY
  PRINT [Please type name to be found]
    PRINT []
  PRINT [and press RETURN.] PRINT []
  MAKE "find FIRST RL
END

TO INKEY
  IF RC = "1 [MAKE "list :name STOP]
  IF RC = "2 [MAKE "list :road STOP]
  IF RC = "3 [MAKE "list :hobby STOP]
  INKEY
END
```

```
TO SEARCH_FOR
  IF :counter > COUNT :list [PRINT SE
    [I don't know] :find TOPLEVEL]
  IF EQUAL? ITEM :counter :list :find
    [STOP]
  MAKE "counter :counter + 1
  SEARCH_FOR
END

TO DISPLAY
  TS
  PRINT ITEM :counter :name
  PRINT SE ITEM :counter :house_no ITEM
    :counter :road
  PRINT SE "Hobby: ITEM :counter :hobby
END
```

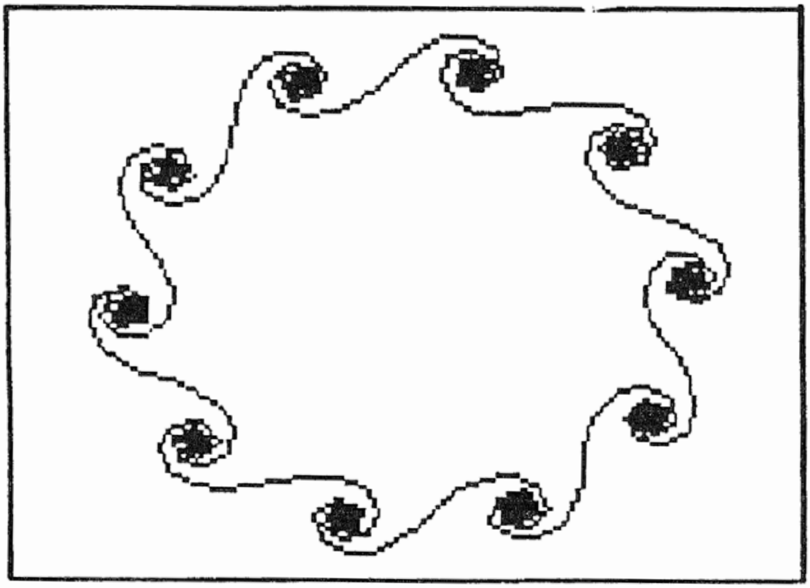
This is a very simple 'starter' database which has a few weaknesses:

1. When searching this database for a road, you must put the entire road name inside square brackets.

2. If there is more than one person with the same name, search_for will only find the first one with that name on the list.

3. If Mary's hobbies were 'swimming and hockey', searching for 'hockey' would not find the record.

To make your own database, decide what information you want, and which parts of it you want to be able to search for.



SQUIGGLE 20 14

```
TO SQUIGGLE :length :angle
FD :length
RT :angle
SQUIGGLE :length (:angle + 10)
END
```

ANAGRAMS

```
TO GO
  PRINT [Give me a word]
  MAKE "word FIRST RL
  TS
  MAKE "anag "
  SHUFFLE :word
  PR :anag
  PR [What was this word?]
  IF EQUAL? :word FIRST RL [PR [CORRECT
    ]] [PR SE [No, it was] :word]
END

TO SHUFFLE :word
  IF EMPTY? :word [STOP]
  REPEAT 1 + RANDOM COUNT :word [MAKE
    "word WORD BF :word FIRST :word]
  MAKE "anag WORD :anag FIRST :word
  SHUFFLE BF :word
END
```

The computer asks for a word, the screen goes blank for a few seconds and the anagram then appears. Ask your partner to see if she or he can solve it.

This is an interesting program to try to understand.

SHUFFLE takes the FIRST letter of the :word and puts it on the end of the :word. It repeats this a random number of times. The FIRST letter is then put in the "anagrams box instead and the remaining letters are used as if they are a new :word.

A murder has been committed. You have to find out who did it, which weapon was used, and where the crime happened.

It will help you to have a written list in front of you:

Suspects	- John	Jack	Mary	Eileen
Weapons	- gun	knife	brick	rope
Rooms	- lounge	bathroom	hall	kitchen

Play with a friend to find who needs fewer guesses.

Suggested alterations:

Change suspects, weapons and rooms. Add motives. Instead of printing WELL DONE, call up a CONGRATULATIONS routine. Add a counter to record how many guesses you needed.

TURTLE IN THE BOX

```
TO GO
  CS PU
  SETPOS [10 10] PD
  SQUARE PU
  SETPOS SE ( 500 - RANDOM 1000 ) ( 300
    - RANDOM 600 )
  MOVE
END

TO SQUARE
  REPEAT 4 [FD 50 RT 90]
END

TO MOVE
  PR [Give me instructions]
  RUN RL
  DETECT
  MOVE
END

TO DETECT
  IF ( AND XCOR > 10 XCOR < 60 YCOR > 10
      YCOR < 60 ) [FOUND] [PR [NOT FOUND]]
  WAIT 20
END

TO FOUND
  PR [FOUND]
  REPEAT 2 [SOUND 1 -15 100 10 SOUND
    1 -15 84 10]
  GO
END
```

A square is drawn and the turtle is positioned at random on the screen.

You must then type in instructions, in LOGO (turtle talk!), to try to guide the turtle into the box. (For example, FD 20 RT 90 FD 60 etc.) If you play the game with a partner, take turns at moving the turtle.

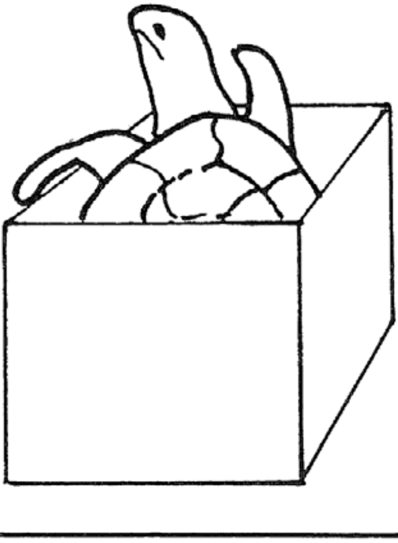
Please notice that the game will stop if you give an incorrect command. You will then have to restart by typing GO.

Suggested alternatives:

1. Draw more than one box.

2. See if you can position the box at random. (This is quite a challenge!)

3. Add a NOT_FOUND routine.



MOONLANDER GAME

```
TO GO
  FENCE CLEAN
  MAKE "base ( 400 - RANDOM 800 )
  PU SETPOS SE :base -300
  SETH 90
  PD REPEAT 2 [FD 10 LT 90 FD 2 LT 90]
  PU SETPOS SE ( 400 - RANDOM 800 )
    ( 300 - RANDOM 200 )
  PR [5 = left 8 = right 0 = boost]
  MAKE "xvel .1 MAKE "yvel 0
  MOVE
END

TO MOVE
  COMMAND
  SETX XCOR + :xvel SETY YCOR + :yvel
  IF YCOR < -265 [RESULT STOP]
  MOVE
END

TO COMMAND
  IF KEY? [MAKE "key RC] [PU STOP]
  IF :key = "8 [RT 30 STOP]
  IF :key = "5 [LT 30 STOP]
  IF :key = "0 [PD MAKE "xvel :xvel +
    SIN HEADING MAKE "yvel :yvel + COS
    HEADING]
END

TO RESULT
  IF AND XCOR + 5 > :base XCOR - 15 <
    :base [PR [WELL DONE]] [PR [YOU MISSED]]
END
```

The TURTLE is a rocket-ship. Can you control it to land on the launch pad?

5 turns L, 8 turns R, and 0 gives a rocket boost.

The remainder of this booklet is concerned with the idea of COMMUNICATING.

Communication is the passing of information from one person to another.

You can use these programs to tell other people about yourself or your school. There are many different ideas here: a program to draw a plan of the classroom, an electronic magazine, a noticeboard, a program to send messages in code, and a program to draw an outlne of your head.

When added to the simple database found near the beginning of the booklet, you now have at least half a dozen different LOGO programs that can be used to pass on information.

But as with the earlier programs, the main idea is for you to experiment.

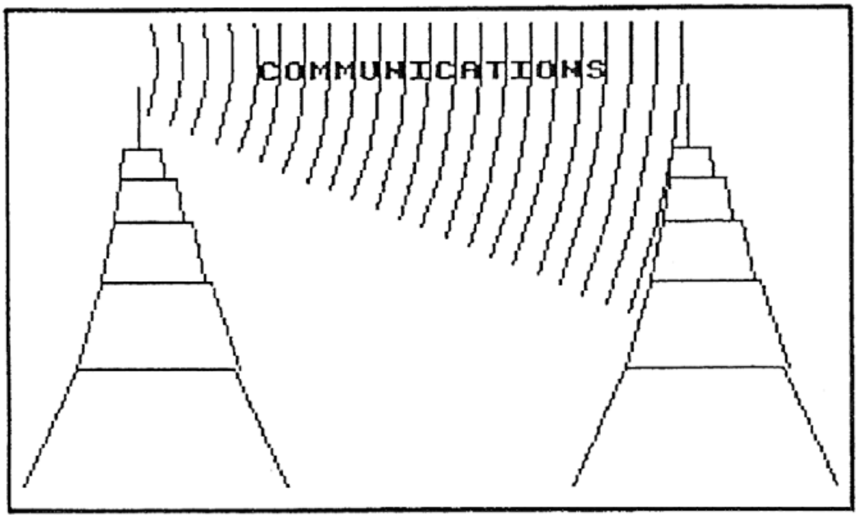
Find out how the programs work.

Then change them to meet your own interests.

Happy programming!

COMMUNICATING

USING L O G O



(The program to draw this picture
can be found on the next page.)

```
TO RING
   REPEAT 8 [SOUND 1 -15 200 8
      SOUND 1 0 200 2
      SOUND 1 -15 200 8
      SOUND 1 0 200 22]
END
```

(Is it a telephone?)

TRANSMISSION TOWER

```
TO GO
  HT WINDOW
  PU SETPOS [-600 -350] PD
  TOWER 200 MAST
  PU SETPOS [100 -350] PD
  TOWER 200 MAST
  RAYS -440 240 24
  VDU [5] PU SETPOS [-300 320] PD
  PR [COMMUNICATIONS\ ] RING
  RECYCLE VDU [4]
END

TO TOWER :s
  IF :s < 40 [STOP]
  SETH :s / 10 FD :s
  MAKE "store POS
  SETH 90 FD :s
  SETH ( 180 - :s / 10 ) FD :s
  PU SETPOS :store PD
  TOWER ( :s * .7 )
END

TO MAST
  SETX ( FIRST :store ) + 20
  SETH 0 FD 100
END

TO RAYS :x :y :z
  IF :x > 180 [STOP]
  PU SETPOS SE :x :y SETH 15
  PD REPEAT 6 [FD :z LT 5]
  RAYS ( :x + 29.2 ) ( :y - 15 )
    ( :z + 5 )
END
```

Use the **RING** routine from the previous page.

SEATING PLAN

```
TO GO
  HT SETUP
  DRAW_DESKS :x :y
  PLACES
  CONTINUE
END

TO CONTINUE
  ASK
  SEARCH :class
  DISPLAY
  CONTINUE
END

TO SETUP
  MAKE "x [-400 -280 -400 -280]
  MAKE "y [120 120 -80 -80]
  MAKE "place [0 0]
END

TO DRAW_DESKS :x :y
  IF EMPTY? :x [STOP]
  PU SETPOS SE FIRST :x FIRST :y
  PD DESK
  DRAW_DESKS BF :x BF :y
END

TO DESK
  REPEAT 2 [FD 160 RT 90 FD 80 RT 90]
END

TO PLACES
  MAKE "class [[[John] [-360 0]]
    [[Mary] [-360 200]] [[Susan]
    [-240 0]] [[Terry] [-240 200]]]
END

TO ASK
  PR [Who do you want to find?]
  MAKE "name RL
END
```

Example:

x	y
-400	120
-280	120
-400	-80
-280	-80

Put these numbers
in the SET_UP routine -

```
MAKE "x [-400 -280 -400 -280]
MAKE "y [120 120 -80 -80]
```

Now consider the :class list.

Think of this like a box of reminder cards.
Each child in the :class has a card wth two
labels on it. The FIRST has the :name on it,
and the LAST has the :place on it.

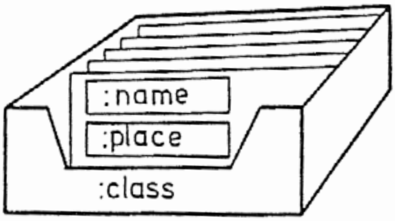
The SEARCH routine works like this:

The FIRST item on the FIRST card is looked
at. IF it is the same as (EQUAL to) the :name
being looked for, the "place is read.
Otherwise the top card is taken away and the
FIRST item on the new FIRST card compared.
IF all the cards are taken away, so the
:class box is EMPTY, then that :name is not
in the class.

MAKE your own "class box in the PLACES
routine.
Work out the coordinates of the middle of
each person's desk for the :place label.

The following line shows how two 'cards' are
put in the 'box' in the program;

```
MAKE "class [[[John ][-360 0]][[Mary ][-360 200]]]
               FIRST    LAST        :name    :place
               label    label       label    label
                 FIRST CARD              CARD
                             :class BOX
```



CODES

```
TO ENCODE :word
  INPUT :word
  CODING :word
END

TO CODING :word
  IF :counter = COUNT :word [PR SE
    [In your code, that is] :word STOP]
  MAKE "letter ( ASCII FIRST :word ) +
    FIRST :pad
  IF :letter > 90 [MAKE "letter :letter
    - 26]
  MAKE "word WORD :word CHAR :letter
  MAKE "pad BF :pad
  MAKE "counter :counter + 1
  CODING BF :word
END

TO DECODE :word
  INPUT :word
  DECODING :word
END

TO DECODING :word
  IF :counter = COUNT :word [PR SE [The
    word was] :word STOP]
  MAKE "letter ( ASCII FIRST :word ) -
    FIRST :pad
  IF :letter < 65 [MAKE "letter :letter
    + 26]
  MAKE "word WORD :word CHAR :letter
  MAKE "pad BF :pad
  MAKE "counter :counter + 1
  DECODING BF :word
END
```

```
TO INPUT :word
  MAKE "pad "
  PR [PLEASE ENTER YOUR 4\-FIGURE CODE]
  MAKE "code FIRST RL
  CHECK :word
  MAKE "repeats 1 + QUOT COUNT :word 4
  REPEAT :repeats [MAKE "pad WORD :pad
    :code]
  MAKE "counter 0
END

TO CHECK :word
  IF NOT NUMBER? :code [PR [It must be
    a number!] TOPLEVEL]
  IF NOT EQUAL? COUNT :code 4 [PR [It
    must have 4 figures.] TOPLEVEL]
END
```

You have to encode or decode a message a word at a time. You need a 4 - figure "password" to code and decode each word. It must be the same 4-figure code to decode the word that was used when it was encoded.

Try: ENCODE "APPLE
and give the 4 - figure code as 1234.
After a few seconds you should get BRSPF.

If you now try DECODE "BRSPF, you will only get APPLE if you use the code 1234.

NOTE: Use CAPITAL LETTERS for words you wish to ENCODE or DECODE.

The next two programs allow you to create a kind of class 'newspaper'.

It is up to you to choose what you want to put on the pages. You can create as many as you wish,-- for news, short stories, poems, facts, jokes, pictures and even music.

In the first program, NOTICES, the pages are shown on the screen in the order in which they are put in the program. The number of REPEATS in the GO routine must be the same as the number of pages you create.

The two lines which begin with SETCUR in the DISPLAY routine may be replaced with WAIT 600 if you prefer to have the pages 'turn' automatically.

In the second program, MAGAZINE, you can turn to any page you wish from an index which is printed first. You do not have to have 6 pages, as in the example, but you must use letters for each page.

You will need the GO and SELECT routines. Alter GO to fit the particular magazine that you create, but leave SELECT as it is. Its job is to find and RUN the chosen page, or to take you back to the index if you press the wrong key.

You will notice that page E calls up another routine (TUNE). There are many ways in which pages can call up extra routines. You may like to invent some.

NOTICES

```
TO GO
  MAKE "number 0
  REPEAT 3 [DISPLAY]
  GO
END

TO DISPLAY
  TS
  MAKE "number :number + 1
  RUN SE WORD "PAGE :number []
  SETCURSOR [25 0] PR SE "PAGE :number
  SETCURSOR [2 21] TYPE [PRESS ANY KEY
    FOR NEXT PAGE] PR RC
END

TO PAGE1
  PR [WELCOME TO]
  PR [] PR [THE NOTICEBOARD]
  PR [] PR [] PR [The following pages
    will tell]
  PR [you something about the school.]
END

TO PAGE2
  SETCURSOR [4 6] PR [This shows ...]
  SETCURSOR [4 8] PR [we can print]
  SETCURSOR [0 12] PR [wherever we want
    to]
  SETCURSOR [18 13] PR [on the screen.]
END

TO PAGE3
  CS HT
  REPEAT 6 [REPEAT 6 [FD 160 RT 60] FD
    40 RT 60]
  SETCURSOR [1 5] PR [We can also have
    pictures!]
END
```

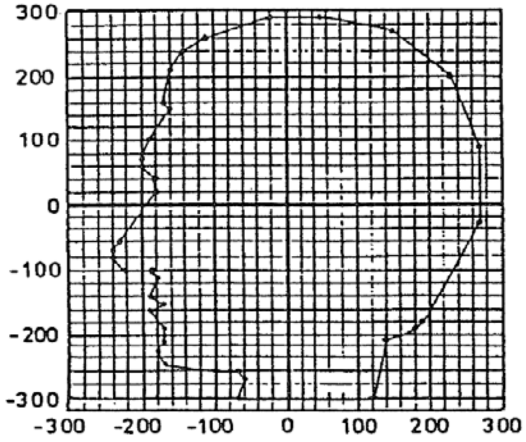
MAGAZINE

```
TO GO
  TS
  SETCURSOR [10 0] PR [MAGAZINE]
  PR [] PR [On the following pages:]
  PR [] PR [] PR [Page A ... Football news]
  PR [] PR [Page B ... Class 4's outing]
  PR [] PR [Page C ... A poem by June]
  PR [] PR [Page D ... A pattern by Terry]
  PR [] PR [Page E ... A tune by Jonathan]
  PR [] PR [Page F ... Jokes page]
  SETCURSOR [2 20] PR [PRESS LETTER of page
    wanted.]
  SELECT RC
  SETCURSOR [5 21] TYPE [PRESS ANY KEY TO
    CONTINUE] PR RC
  GO
END

TO SELECT :letter
  IF NUMBER? :letter [GO]
  IF DEFINED? :letter [TS RUN SE :letter []]
    [GO]
END

TO A
  SETCURSOR [8 0] PR [FOOTBALL NEWS]
  SETCURSOR [0 4] PR [In the home match
    against]
  PR [West Hill United last Tuesday.]
  PR [Gary scored a magnificent goal]
  PR [from a penalty in the closing]
  PR [minutes to win us the match]
  PR [by 1 goal to nil.]
END
```

```
TO B
  SETCURSOR [8 0] PR [CLASS 4 OUTING]
  SETCURSOR [0 4] PR [Last Wednesday, we went
    to the]
  PR [zoo.]
  PR [] PR [We had a great time.]
  PR [Scamp forgot his sandwiches]
  PR [but the chimpanzees offered]
  PR [him a banana.]
  PR [] PR [We said Scamp should be in the]
  PR [chimpanzees' cage!]
END

TO C
  SETCURSOR [3 0] PR [POEM by June Day]
  SETCURSOR [2 4] PR [My little dog]
  SETCURSOR [4 6] PR [Behaves like a hog!]
  SETCURSOR [2 8] PR [He eats and he eats
    and]
  SETCURSOR [23 9] PR [he eats!]
  SETCURSOR [4 11] PR [Not only his meat]
  SETCURSOR [2 13] PR [But biscuits of wheat]
  SETCURSOR [4 15] PR [And worst of all ...]
    WAIT 200
  SETCURSOR [6 18] PR [ALL OF MY SWEETS!]
END

TO D
  CS HT
  REPEAT 6 [REPEAT 6 [FD 160 RT 60] FD 80
    RT 60]
  SETCURSOR [1 0] PR [PATTERN by Terry Way]
END
```

```
TO E
  PR [Do you like my tune?]
  PR [] PR [By Jon]
  WAIT 50
  TUNE
END

TO TUNE
  REPEAT 2 [SOUND 1 -15 100 10 SOUND 1 -15
    92 10 SOUND 1 -15 84 20]
END

TO F
  SETCURSOR [8 0] PR [JOKES PAGE]
  SETCURSOR [0 8] PR [What does a cowboy wear
    at]
  PR [] PR [the North Pole?]
  SETCURSOR [5 21] TYPE [Press any key]
  PR RC
  SETCURSOR [3 13] PR [A wild vest!]
END
```

OUTLINE



Make a shadow graph of your head on a large sheet of graph paper. Scale it to produce x and y coordinates to fit the TURTLE screen. (You may need extra help for this.) Put the coordinates into lists as shown.

x	y	x	y	x	y	x	y
120	-300	-110	260	-180	20	-190	-160
140	-210	-140	240	-230	-60	-170	-190
180	-190	-160	210	-240	-70	-170	-210
190	-180	-170	160	-240	-80	-180	-230
270	-30	-160	150	-220	-100	-170	-250
270	90	-190	100	-190	-100	-100	-260
230	200	-200	70	-180	-110	-70	-260
150	270	-200	60	-190	-140	-60	-270
50	290	-180	40	-170	-150	-70	-300
-20	290						

```
TO MIKE
  MAKE "x [120 140 180 190 270 270 230
     150 50 -20 -110 -140 -160 -170 -160
     -190 -200 -200 -180 -180 -230 -240
     -240 -220 -190 -180 -190 -170 -190
     -170 -170 -180 -170 -100 -70 -60 -70]
  MAKE "y [-300 -210 -190 -180 -30 90
     200 270 290 290 260 240 210 160 150
     100 70 60 40 20 -60 -70 -80 -100
     -100 -110 -140 -150 -160 -190 -210
     -230 -250 -260 -260 -270 -300]
  HT PU SETPOS SE FIRST :x FIRST :y
  PD DRAW :x :y
  SETPOS SE FIRST :x FIRST :y
END

TO DRAW :x :y
  IF EMPTY? :x [STOP]
  SETPOS SE FIRST :x FIRST :y
  DRAW BF :x BF :y
END
```