LOGO

GRAPHICS INTERPRETER

Kuma

AMSTRAD

CPC-464

LOGO

FOR THE

AMSTRAD CPC 464 MICROCOMPUTER

ISBN NO Ø7457-ØØØ9-8

Published by:- Kuma Computers Ltd.,
12 Horseshoe Park,
Pangbourne,
Berks RG8 7JW

Telex 849462 Tel Ø7357 4335

LOGO TURTLE-GRAPHICS INTERPRETER

by

S.J.Wainwright. 1984.

CONTENTS.

LOGO TURTLE-GRAPHICS Interpreter.

Introduction.

LOGO Graphics is a tool for practical exploratory geometry. The control of the Turtle on the screen provides experience of lengths, directions, angles, radii, and colour. The user can discover the structure of geometric shapes, he/she can experiment with patterns, and can gain invaluable experience in the estimation of quantities such as distance and angles. By interacting with the computer, the user gains confidence in the use of the computer as a tool, and can use it for example, to discover alternative solutions to particular problems.

The use of Turtle Graphics stimulates mathematical thinking and discussion among users, and is particularly useful for group work. The idea of programming can be introduced in the context of 'teaching' the computer new procedures which, when 'learned' by the computer can be used to perform complicated sets of actions. Programming with LOGO Graphics is a creative activity which can be pursued by all ages, and with mixed age groups. New procedures can be saved on tape for future use. Calculator procedures are included to free the user from the use of other calculating aids.

The Turtle.

The Turtle exists on the screen as a round object with a direction pointer. By the use of various commands, the Turtle can be made to face in any direction, to move around the screen either leaving a trail as it goes, erasing a previously drawn trail, or jumping from one place to another without leaving a trail. The Turtle can also draw circles around itself or fill in discs of various colours, and draw lines of various colours. The Turtle can be either visible or invisible. If it is visible, then the process of drawing lines etc. is slower than if the turtle is invisible.

Text messages can be displayed at the top of the screen and are called from within user defined procedures. Thus, if the Turtle draws a house for example, the text message: "House" could be displayed at the top of the screen if required.

The LOGO-TURTLE-GRAPHICS implementation.

Getting Started

Load the Interpreter by placing the tape in the cassette recorder, and typing ' chain"logo" '. The Interpreter will load and then auto run. You will be asked whether you want m or l resolution. It is best to respond with m unless you wish to use several colours. If you respond with l , the drawings produced will be at a lower resolution, but 16 colours will be available.

Procedures.

This implementation of LOGO TURTLE-GRAPHICS has a dictionary containing 44 primary procedures (i.e. supplied in the system). Each procedure has a name e.g. 'forward' or 'left'.

Commands.

A command is given when the name of a procedure is entered (along with a parameter if required). When a command is given, the procedure then executes.

Parameters.

Many of the procedures require parameters, i.e. numbers which must be entered after the name of the procedure. For example, the procedure 'draw' requires a parameter which specifies the length of the line to be drawn.

Immediate mode.

Commands are entered in response to a screen prompt of "Enter procedure name ?". The user then types in the name of the procedure required. If the procedure requires a parameter, then the screen prompt will change to a request for the parameter. For example : When the screen prompt is "Enter procedure name ?", if we enter "draw" then the screen prompt will change to "Distance ?". If we then enter a number, say, 100, the screen Turtle will move forward (i.e. in the direction it is pointing) a distance of 100 screen units, and will draw a line as it goes. The screen is 640 units from left to right, and 400 units from top to bottom.

Some procedures e.g. 'clear' do not require a parameter. In these cases, when the name of the procedure is entered in immediate mode, it will execute immediately.

In procedure definitions however, the name of every procedure must be followed by a comma and then a parameter value. e.g. "draw,100". This is true even if the procedure does not require a parameter. In these cases the parameter is ignored and it is often convenient to use a zero as a dummy parameter, e.g. "clear,0". This is discussed later under procedure definitions.

Primary procedures available.

Many of the primary procedures have abbreviations which can be freely interchanged with the full procedure name. These abbreviations are shown below in parentheses, after the full procedure name. If a procedure is only available in immediate mode, then this is indicated below by (IMO). If on the other hand, a procedure is available only within a user defined procedure, then this is indicated by (UDPO).

hideturtle (ht) Causes the Turtle to become invisible. All movements and line drawings are much quicker when the Turtle is invisible.

showturtle (st) Causes the Turtle to become visible. This slows down the process of moving around the screen and drawing lines. It is best to have a visible Turtle whilst working in immediate mode because one can see every movement of the Turtle. It is usual to work out the details of movements which are to be incorporated into user defined procedures, by using a visible Turtle. (It is also good practice to make notes whilst experimenting with a visible Turtle in this way.). When user defined procedures are executed however, the Turtle should be hidden so that a fast execution of the procedure will be obtained. The Turtle can be hidden either immediately before a procedure is called, or the command "ht,0" can be used as the first command of the user defined procedure.

penup (pu) Lifts a notional pen off the screen. If the Turtle is then moved forward by the procedure 'forward', it will move without drawing a line.

pendown (pd) Places a notional pen down on the screen. If the Turtle is moved by the procedure 'forward', it will move and draw a line as it goes.

forward (fd)
n Causes the Turtle to move forward (i.e. in the direction it is pointing) a distance of n units. Depending on the current penup or pendown condition, the Turtle will move without or with drawing a line.

reverse (rvs) n	Causes the Turtle to move backwards a distance of n units, drawing a line as it goes. The Turtle remains facing in the same direction that it was before the 'reverse' command was given.
draw (dr) n	Causes the Turtle to move forward a distance of n units, drawing a line as it goes.
back (bk) n	Causes the Turtle to move backwards a distance of n units, erasing any previously drawn line as it goes. The Turtle remains facing in the same direction that it was before the 'back' command was given.
erase (er) n	Causes the Turtle to move forward a distance of n units, erasing any previously drawn line as it goes.
jump (jp) n	Causes the Turtle to jump forward a distance of n units, without drawing a line, and without erasing anything in its path.
left (lt) n	Causes the Turtle to turn left (with respect to the direction it is currently pointing) through n degrees.
right (rt) n	Causes the Turtle to turn right through n degrees.
circle (cr) n	Causes a circle of radius n units to be drawn with the Turtle at its centre.
fill (fl) n	Causes a solid disc of radius n units to be drawn with the Turtle at its centre.

colour (col)
n
Causes any subsequent lines, circles or discs drawn by the Turtle to be of the colour indicated by n, where n is the normal Amstrad colour code. The colours available will depend on the resolution selected on loading the Interpreter.

clear (cl) Clears the screen, and re-centers the Turtle facing right.

home (h) Leaves the screen display intact, but re centers the Turtle facing right.

pause (ps) Causes a pause of n seconds.
n

size (si)
n
Adds n to a size-controlling variable. The value of the size-controlling variable is automatically added to the parameter of the 'draw' procedure, or the 'forward' procedure if pendown is true, before 'draw' or 'forward' is executed. The size-controlling variable is initialised to zero on loading the Interpreter, and by the use of the procedure 'reset'. 'size' can be used to pass parameters to the 'draw' procedure or the 'forward' procedure, even within user defined procedures. This means that generalised procedures can be defined with all of their 'draw' or 'forward' procedures having a parameter of zero. The size of the object to be drawn can then be controlled by the use of 'size' before the procedure is called. As a simple example :

```
size
100
draw
0
```

will draw a line 100 units long.

It must be remembered to reset the size-controlling variable to zero by using the procedure 'reset' after 'size' has been used, otherwise the Turtle will not draw lines of the intended length.

angle (an)
n

This procedure is analagous to 'size' but it adds n to an angle-controlling variable. The value of the angle-controlling variable is automatically added to the parameters of the procedures 'left' or 'right' before they are executed. The angle-controlling variable is initialised to zero on loading the Interpreter, and by the use of the procedure 'reset'. 'angle' can be used to pass parameters to the procedures 'left' and 'right' even within user defined procedures. As a simple example :

```
angle
45
right
0
```

will cause the Turtle to turn right through 45 degrees. It must be remembered to reset the angle-controlling variable to zero after 'angle' has been used.

reset (re) Resets the size-controlling and angle-controlling variables to zero.

repeat (rpt)
n

The procedure 'repeat' is used in conjunction with the procedure 'continue' to repeat a sequence of <u>primary procedures</u> n times. 'repeat'-'continue' loops can be nested two deep. In immediate mode, when 'repeat' is entered, the screen prompt changes to "number of times ?". Then n, the number of times is entered. After this point, all of the subsequent commands must be entered as a procedure name followed by a comma and a parameter value (or dummy parameter).

For example :

```
repeat
2
repeat,3
draw,60
left,120
continue,0
jump,80
continue,0
```

Or, the same thing using abbreviations :

```
rpt
2
rpt,3
dr,60
lt,120
cont,0
jp,80
cont,0
```

continue (cont) Used in conjunction with repeat as above.

moveto (mt) (IMO)
x,y The procedure 'moveto' takes two parameters, x and y, which are horizontal and vertical coordinates on the screen. They are entered, separated by a comma. When 'moveto' executes, the Turtle is moved to the position specified by the two coordinates.

The definition of User defined procedures by using the procedure 'to'

When the command 'to' is entered, the Interpreter goes into compile mode and asks for the name of the procedure which is to be defined. Then, the commands that make up the definition of the new procedure are entered in sequence. Each command is entered as a procedure name, a comma, and a parameter value. If any procedure within the definition does not require a parameter, then a dummy parameter <u>must</u> be given. Zero is a convenient dummy parameter. The procedure definition is terminated by entering 'end,0'.

New procedures are compiled into the dictionary, and are executed whenever their names are entered.

A user defined procedure is a <u>First generation</u> procedure if <u>primary</u> procedures <u>only</u> are used in its definition.

Below are given two examples of first generation procedures called 'case' and 'face'. These two first generation procedures will then be incorporated into the definition of a second generation procedure called 'clock'

```
to                            to
case                          face
ht,0                          ht,0
jp,100                        rpt,12
rt,90                         jp,80
jp,100                        rvs,20
rpt,4                         bk,60
rt,90                         lt,30
dr,200                        cont,0
cont,0                        cr,90
h,0                           rpt,48
end,0                         dr,60
                              bk,60
                              rt,30
                              cont,0
                              dr,60
                              h,0
                              end,0
```

It should be noted in the above examples, that although 'ht','h','cont', and 'end' do not require parameters, they had to be entered with dummy parameters (0) in the procedure definition.

A user defined procedure is a <u>Second generation</u> procedure if it contains one or more first generation procedures in its definition. For example :

```
to
clock
case,0
lt,90
jp,110
fl,10
h,0
face,0
end,0
```

The procedure 'clock' is a second generation procedure because it contains the first generation procedures 'case' and 'face' in its definition. It should be noted that primary procedures and first generation procedures can be mixed in the definition of second generation procedures. Second generation procedures cannot be used in the definition of further procedures. It should be noted that the first generation procedures 'case' and 'face' were both entered with the dummy parameter '0', because the user defined procedures do not use parameters directly. The user should define the procedures 'case' and 'face' and then 'clock' as above, and then type in 'clock'. A clock will be drawn on the screen, and a clock hand will move around the clock face.

edit (ed) (IMO)

The Editor is called by typing in 'edit'. The Editor is a device which allows simple changes to be made to procedure definitions after they have been compiled into the dictionary. When the Editor is called by entering 'edit', the user is asked to enter the name of the procedure to be edited. When the name of the procedure is entered, the definition of the procedure is given on the screen, with a number preceding each command. The user has the option to quit or to proceed with an edit. If the user chooses to edit the procedure, he/she is then asked for the number of the command to be replaced, and for the replacement command; which consists of a procedure name, a comma, and a parameter (or dummy parameter). The user then has the option of editing another line or of quitting the Editor.

As the Editor does not allow the insertion of lines, it is best to include one or two dummy commands before 'end,0' when a procedure is defined. An example of a dummy command would be :

'dummy,0'

The Interpreter ignores unrecognised commands, so the dummy commands will not affect the operation of the defined procedure. They will, however, reserve space for any insertions which may be required at a later date. Dummy commands could also be used as comments as long as they had the correct format.

text (IMO) Allows a text message to be written into the text file. The text file can hold up to 15 messages.

Text messages can be called by user defined procedures which cause the text message to be printed at the top of the screen.

When the command 'text' is entered in immediate mode, the computer asks for the number of the message to be written. This number can be from 1 to 15. When the number is entered, the computer then asks for the message to be entered. The message can be up to 2 lines in length.

viewtext (vt) (IMO) Displays all of the messages in the text file, on the screen.

cleartext (clt) (UDPO) Clears the text message display.

showtext (sh) (UDPO)
n

Causes text message numuer n to be displayed at the top of the screen. This procedure is only availaple from within user defined procedures. For example, suppose that text message 1 had been entered as 'A working clock' by the use of the procedure 'text'. It would then pe possible to display this message during the execution of the user defined procedure 'clock' as previously defined, by making the first command of the procedure 'clock' pe : 'sh,1'.

Programming a repeating sequence of user defined procedures by using the procedure 'prog'.

The procedure 'prog' is used to program a repeating sequence of user defined procedures which have been compiled into the dictionary. 'prog' behaves rather like 'repeat', but only uses user defined procedures, and no parameters.

prog (IMO)
n

The procedure 'prog' is used in conjunction with the procedure 'execute', to repeat a sequence of user defined procedures n times (where n can equal 1). For example, suppose that we had previously defined the procedures 'triangle','ball' and 'leap' as follows :

```
to              to          to
triangle        pall        leap
rpt,3           fl,20       jp,90
dr,60           end,0       end,0
lt,120
cont,0
end,0
```

We could use 'prog' to draw a triangle, a ball, another triangle, and another pall, as follows :

```
prog
2
triangle
leap
ball
leap
execute
```

execute (ex) (IMO) Used in conjunction with 'prog' as above.

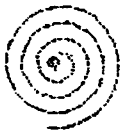
Using 'size' and 'angle' within procedure definitions.

The procedures 'size' and 'angle' can be used to change progressively the parameters of the procedures 'draw', 'forward' when 'pendown' is true, and 'left' & 'right' respectively, as follows :

Three examples :

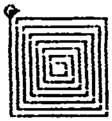
```
to
spiral
rpt,100
dr,20
lt,15
size,-0.2
cont,0
reset,0
end,0
```

Resulting drawing.



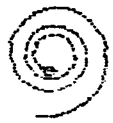
```
to
grow
rpt,50
dr,10
rt,90
size,4
cont,0
reset,0
end,0
```

Resulting drawing.



```
to
snail
rpt,80
dr,20
lt,15
size,-0.2
angle,0.25
cont,0
reset,0
end,0
```

Resulting drawing.



Saving user defined procedures and text messages on Tape files, or loading them from tape by using the procedure 'sys'.

sys (IMO)

If the command 'sys' is entered, the user is given the option of downloading previously recorded procedures and text messages from a named tape file, or of saving new user defined procedures and text messages on tape in a named tape file. Once the appropriate commands have been given, the computer takes several seconds to organise the files, and the user should wait for the screen prompts.

The Calculator procedures. (IMO)

add
n1,n2 Adds n1 to n2.

With the calculator procedures, after the name of the procedure has been entered, e.g. 'add', the computer prints out the name of the procedure at the bottom of the screen. The user then enters the two parameters, separated by a comma. The computer then prints out the answer at the top of the screen. The user is then given the option of leaving the answer on display, or of removing the answer.

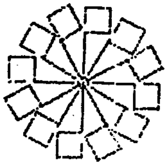
subtract (sub) n1,n2	Subtracts n1 from n2.
multiply (mult) n1,n2	Multiplies n1 by n2.
divide (div) n1,n2	Divides n1 by n2.
remainder (rem) n1,n2	Returns the remainder of n1/n2.
quotient (quo) n1,n2	Returns the quotient of n1/n2
raise n1,n2	Raises n1 to the power n2. It should be noted that the procedure 'raise' can be used for example, to obtain square roots, e.g.: raise 16,0.5 will return an answer of 4.
procs (IMO)	Lists the names of the user defined procedures.
menu (IMO)	Displays the names and abbreviations of all 44 primary procedures.
stop (IMO)	Stops the Interpreter and returns the computer to BASIC.

<u>**Examples.**</u>

The following are four procedure definitions which illustrate just some of the results which can be achieved with the LOGO TURTLE GRAPHICS INTERPRETER :

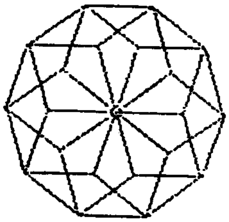
```
to
wheel
repeat,12
draw,60
repeat,4
draw,30
right,90
continue,0
reverse,60
right,30
continue,0
end,0
```

<u>Resulting drawing</u>.



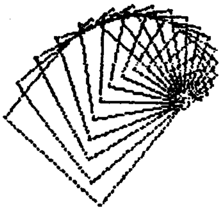
```
to
diamond
repeat,10
right,36
repeat,5
draw,80
right,72
continue,0
continue,0
end,0
```

<u>Resulting drawing</u>.



```
to
fan
repeat,15
repeat,4
draw,10
left,90
continue,0
left,10
size,10
continue,0
reset,0
home,0
end,0
```

<u>Resulting drawing</u>.



```
to
star
repeat,12
repeat,4
draw,80
left,90
continue,0
left,30
continue,0
end,0
```

<u>Resulting drawing.</u>