IBCS

Drawing with BASIC

Mr. Brennan

This Lab is an OPTIONAL extension of BASIC Programming
You may submit programs that you create with the graphics and drawing statements for some extra credit.

Part I BASIC Graphic Mode and Drawing

Description: Create a new window for program output, explore the drawing commands of BASIC.

Deliverable: Optional - programs can be submitted via google drive.

DRAW str$
Statement. Draws the object described by a string "str$" of graphics language commands in the current graphics mode (see MODE). The following table details the graphics command language:

[See also Blassic Reference]

Command	Description
B	Causes the cursor to not DRAW on the next motion command
C n	Changes the current drawing color to value "n" (seeCOLOR for table of valid color values for "n")
U n	Moves up "n" pixels from the current position
D n	Moves down "n" pixels from the current position
L n	Moves left "n" pixels from the current position
R n	Moves right "n" pixels from the current position
E n	Moves diagonally up and right "n" pixels from the current position
F n	Moves diagonally down and right "n" pixels from the current position
G n	Moves diagonally down and left "n" pixels from the current position
H n	Moves diagonally up and left "n" pixels from the current position
M x, y	Moves to the coordinate specified by "x" and "y". Motion isabsolute unless "x" and/or "y" are prefixed with either "+" or "-", in whichcase the move will be relative to the current position.

IBCS

Drawing with BASIC

Part I

The MODE statement below creates a graphics window that is 400 pixels by 400 pixels.

```
MODE  400, 400 :
```

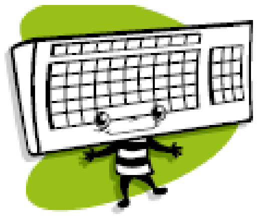
The statement below draws a square at coordinates (100,100)

```
DRAW "BM100,100;R20;D20;L20;U20"
```

Draw a square by drawing a line of length 20 to the right, then a line of length 20 down, then 20 left, then 20 up,

M means move - M100,100 means move to 100,100

"B" means to not draw lines while doing the next move



1. Create a new program with the following three lines to draw a square in a new window:

```
MODE  400, 400 :
DRAW "BM100,100;R20;D20;L20;U20"
INPUT A$
```

Without the INPUT A$ the program would end and the new window would close; the INPUT statement just waits for you to hit enter before terminating the program.

2. Test your program. Notice that the program opens a new window with a white background. There are lots of things you can do with the BASIC graphics.

Diagonal Lines

Suppose you wanted to draw a line from A to B (fig 1.)

E n	Moves diagonally up and right "n" pixels from the current position
F n	Moves diagonally down and right "n" pixels from the current position
G n	Moves diagonally down and left "n" pixels from the current position
H n	Moves diagonally up and left "n" pixels from the current position

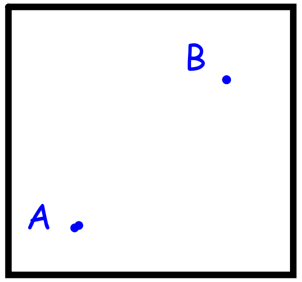


fig 1.

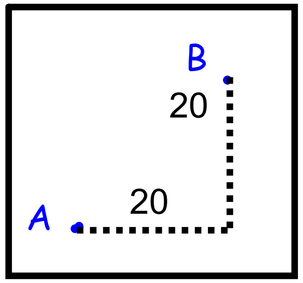
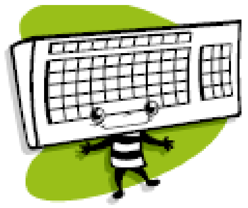


fig 2.

If your current position was at A, then you want E type diagonal of length 20.

Beware that the 20 represents the length right and length up, which is not the length of the line from A to B, sorry Pythagoras.

IBCS



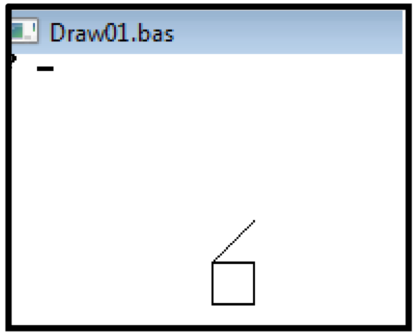
3. Lets start to draw a triangle, which can be a roof for your house. From the previous statements the starting point will be 100,100. Add the next line to draw a diagonal line up to the right 20.

New line →

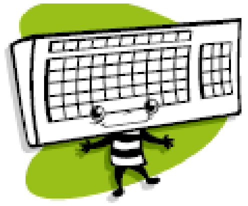
```
MODE  400, 400 :
DRAW "BM100,100;R20;D20;L20;U20"
DRAW "E 20":       REM DRAW A diagonal up to the right
INPUT A$
```

4. Test your program.

5. You can see this is a big roof for a little house. Make the square 40x40.



IBCS

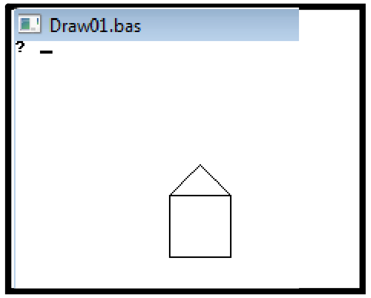


6. Finish the triangle. Here I'll use two separate statements, but they could be combined with the Draw statement that created the first side of the triangle. Also, the third side of the triangle shouldn't be needed because it shares the same side as the square, but at least you will have an example of creating a triangle in case you REMove the square.

New lines

```
MODE  400, 400 :
DRAW "BM100,100;R40;D40;L40;U40"
DRAW "E 20":       REM DRAW A diagonal up to the right
DRAW "F 20":       REM Draw a diagonal down to the right
DRAW "L 40":       REM draw bottom of the triangle
INPUT A$
```

7. Test your program.



IBCS

The rest of the lab I'll leave to you.
Things you could try:

a. Loop for user input - start them at some point on the screen and read user input to draw left, right, down, up, diagonal.

Maybe add commands for "pen up" and "pen down"

b. Experiment with:
DRAWARC x,y,z

c. **GRAPHICS [CLS|PAPER|PEN]**
Statement. Enables the use of the CLS, PAPER, and PEN statements in the graphics modes (see MODE).
COLOR ,0: GRAPHICS CLS

Where CLS will clear whole screen GRAPHICS CLS clears only to current position.

d. See also the PEN, INK, and COLOR statements