CS 151 s10 proj. 1

Philip Prosapio
Charlie Spatz

Code location:pwprosap (Academic Server)- shapeA.py and shapeB.py are included in the single file of shapes.py

The command lists that we used as directors are as followed:

Exercise 1) An object that has 4 rectangles that protrude from a central square.

Exercise 2) An object that consists of four separate triangles that are put together to make one bigger triangle.
Triangle- equilateral triangles that are all the same size.

Exercise 3)
pen down
forward 100
left 90
forward 20
left 90
forward 80
right 90
forward 80
left 90
forward 20
left 90
forward 100
forward 200
left 90
forward 100
left 90
forward 25
left 90
forward 50
right 90
forward 150
right 90
forward 50
left 90
forward 25
left 90
forward 100

Exercise 4)
forward 200
left 90
forward 100
left 90
forward 25
left 90
forward 50
right 90
forward 150
right 90
forward 50
left 90
forward 25
left 90
forward 100

Write Up Responses

1) Yes, the Artist new the meanings of the key words in the description, like rectangle, square and protrude. So it was made easier to draw out the logo or shape.

2) We both had a shared knowledge of what the shapes that were used to describe the logo, so this helped the artist to understand the instructions. But when we look back, we realized that more specific information would be needed to draw the exact picture.

3) We did not give exact lengths of sides or size of the logos. Also terms like rectangles can be construed in different ways. Since a square is technically a form of a rectangle using the two words in the definition could leave the instructions open to interpretation.

4) There was a large amount of ambiguity in our first 2 command sets; we could interpret the commands in multiple different ways. But in the later commands, we needed to be very precise or the computer wouldn't know what to do. Computers cannot handle any ambiguity at all; they need a well-defined straightforward set of instructions.
There were more commands or information given in task 4 than 3 because the figure for task 4 was a little more complicated and had more angles and sides, which required more commands.

5) Yes it helped a lot. Instead of having to retype all of the commands or copy-paste all of them, we could just define the set of commands under one name and just type the name whenever it is needed.

6) If we tasked a set of commands to execute itself, then it would execute itself continuously, which is called recursion. Since every time the set of commands reached the bottom, it will command itself to start over again in a never-ending loop.

7) We made 3 pentagons of differing size with our last version of our shape command list. We could possibly make many more shapes then we did with the first basic set of commands.

8) No, Shape E does not care what list of commands that it called. It will execute any set of commands, no matter what you do to Shape D.

9) Understanding how to use parameters. We also ran into problems when trying to use variables. And also some of our designs involved a lot of math to figure out the angles and side lengths.

Note: We have completed two extensions for this lab. In our shapes.py file we have at the bottom our N-gon code and our complex supershape code. The Ngon code obviously completes the Ngon extension and the complex supershape code completes the extension that asks us to make a code that will draw and interesting graphic.

This is a thumbnail photo of our interesting graphic!!