Lab 4: Turtle Graphics

Due: Sunday, August 4th, 10:00 p.m.

Imagine a robotic turtle on the x-y coordinate plane starting at (0, 0). Say you give it the command turtle.forward(15), and it moves (on-screen!) 15 pixels in the direction it is facing, drawing a line as it moves. Give it the command turtle.left(25), and it rotates in-place 25 degrees clockwise. By combining together these and similar commands, intricate shapes and pictures can easily be drawn.

The turtle module is an extended reimplementation of the same-named module from the Python standard distribution up to version Python 2.5. It tries to keep the merits of the old turtle module and to be (nearly) 100% compatible with it. This lab will enable you to learn some of the basic programming commands, classes and methods of this module, using them interactively to create your own 2D pixel art drawings.

Part 1: Lists (50 pts)

In Python, a list is used to save collection of values. For instance, to save all the different days of the week we could declare a variable called “daysOfTheWeek” and assign a list as follows:

```
daysOfTheWeek = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday",
"Saturday", "Sunday"]
```

Now look at this program used to display all the days of the week:

```
for i in range (0, len(daysOfTheWeek)):
   print daysOfTheWeek[i]
```

We often will use loops to work with lists because it is an efficient way to access and sometimes modify the elements of a list.

Sometimes you need more than one dimension to a list. In Python you can achieve this by creating a **list of lists!**

For instance for our pixel art project we want to store each “pixel” on a line within a list:

```
e.g.:
line1=[1,0,1,0,1,0]
line2=[0,1,0,1,0,1]
line3=[1,0,1,0,1,0]
```

We can then store all these lines into a list. Let’s call this list “grid”:

grid=[line1, line2, line3]

Now we have a list, where each element is itself a list.
A quicker way to do this is to do it all in one line as follows:

grid = [[1,0,1,0,1,0],[0,1,0,1,0,1],[1,0,1,0,1,0]]

We can then access any “cell” within this grid using the following command:

grid[i][j]

Where i is the index of the top level list and j is the index of the inner list at index i.

E.g.
grid[2][4]

This would return the value of the 5th cell at the 3rd line (element) of our grid!

Now, for this lab, we will use a list of lists to store the grid of pixels for a drawing. Each “pixel” is not actually a pixel, rather we will have it be an area of set pixels so we see it better because pixels are too small to see a drawing on that scale. Each “pixel” will be a box of 10x10 pixels each. We will use the turtle module in the python library to then take this grid of pixels and actually draw it out to our screen.

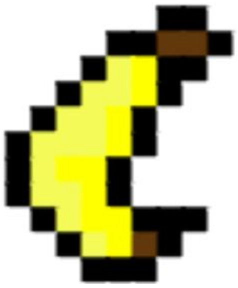
Read through the given main.py file to learn about the functions you were given and how the turtle module works.

For this part of the lab, your only task is to understand lists, understand how the turtle module works and write the draw() function in the main.py file so it can take a list of colors and a grid of pixels to make a drawing.

Testing
You should be writing test functions for each task, ensuring that the game runs correctly.

Part 2 (50 pts): ***What can you draw?***

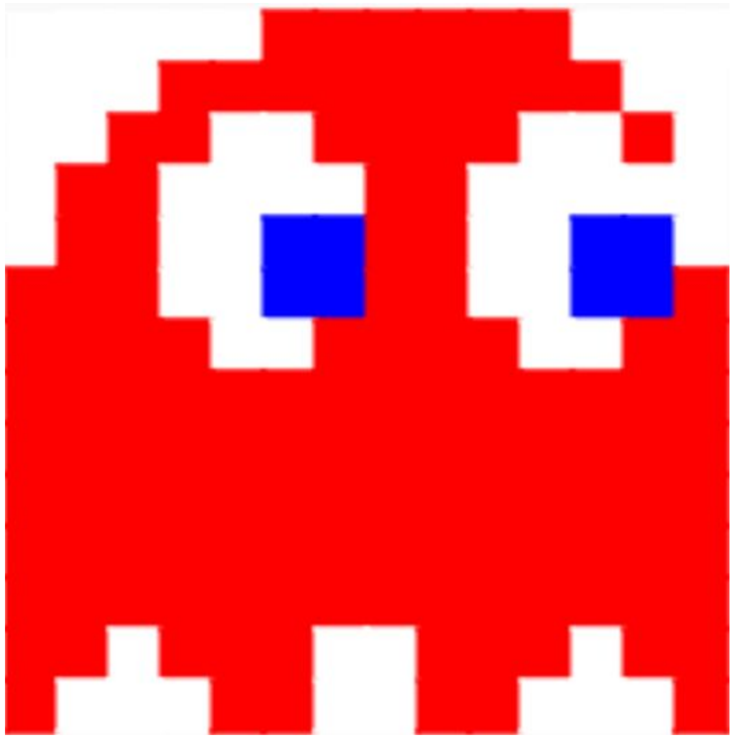
Let's get creative. You saw how I can draw a banana from a grid of pixels:



Now you will be making your own drawings, from the pictures below and from your own imagination.

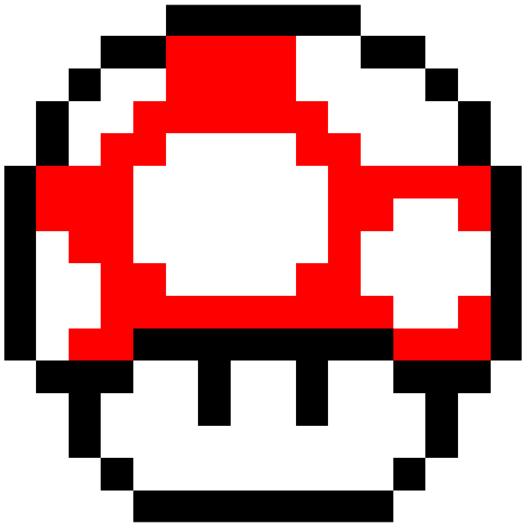
Task:

- Write different palettes and grids to make the following drawings (in colors of your choice):

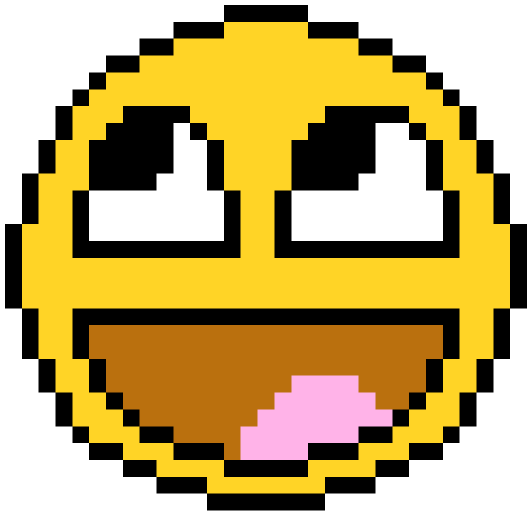
1. 

4.

2.



3.



4.

5. **Two of your own drawings.**

Testing
You should be writing test functions for each task (in a different file, make use of ***import***), ensuring that your program runs correctly.

Challenge (bonus pts)

There are a few challenge functions in the main.py file, each worth two bonus points.

Submit

To submit your work, simply commit and push the files to your lab4 directory in your git repository.