Week 11 Lab Assignment Goals

- Learn how to use Pygame module and draw shapes
- Learn how to design classes and make your program object-oriented

Step 0 : Create a GitHub repository

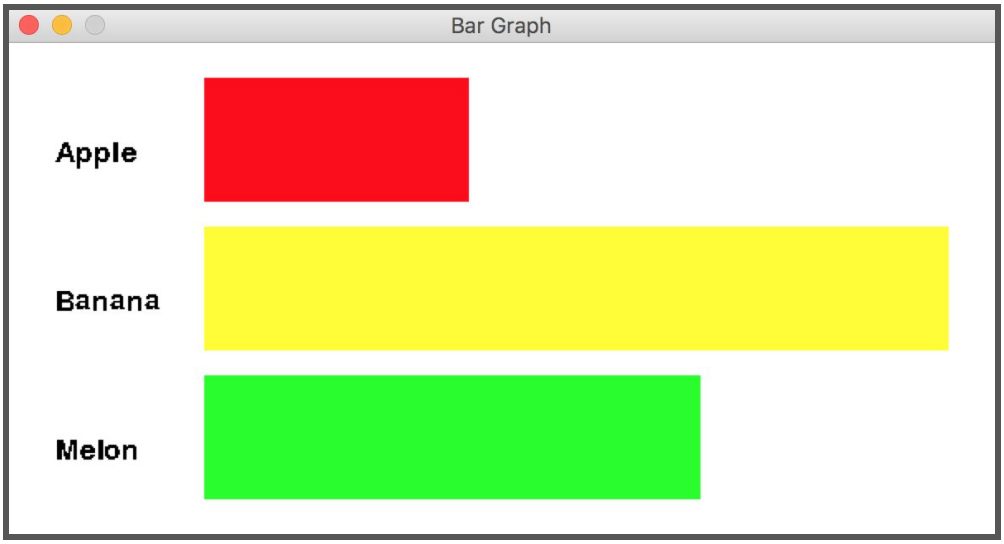
- Go to https://classroom.github.com/assignment-invitations/90160a17ad08e3737f57629b68ac34aa
- Accept the assignment invite and clone the assignment repository onto your machine
- Open a Terminal window or command prompt and 'cd' to the cloned directory

Step 1 : Play with sample codes for better understanding

- Run sample codes respectively (from sample1.py to sample6.py), and try to fully understand the logic of them.
- Sample 1
    - We can get all events happened on the gaming window.
    - The gaming window is getting all events (mouse movement, clicks, keyboard inputs, etc.), and they will be printed out on your console(terminal window).
    - **If you can see mouse movements but not clicks, please let GSIs know.**
    - You won't be able to quit the game by clicking X button on the top left corner of the window (on purpose). Do Ctrl+C on your console.
- Sample 2
    - Since we update 'gameExit' variable to 'True' when we click X button, now you can quit the game normally.
- Sample 3
    - We can fill out different colors to the background.
    - Try to change the background color into blue.
- Sample 4
    - We can draw shapes.
    - In this file, you'll found different methods of drawing different shapes.
    - Refer to the Pygame Documentation for Drawing to figure out what each function parameter means.
    - Try to change the coordinates, color, size of the shapes.
- Sample 5
    - We can make the shapes move.
    - In this file, we're getting keyboard inputs(basically arrows) to make the rectangle move.
- Sample 6
    - We can change the speed of the game and the movement of the shape.
    - We added a string to this file. Refer to the Pygame Documentation for Fonts to figure out how to define a string object and display it.
    - Try to understand the differences from sample 5. How does sample 6 code updates coordinates of the rectangle? Why are you able to move the rectangle without keeping putting keyboard inputs?
    - Try to change the number in clock.tick() function. It will change the speed of the game.

Step 2 : Draw bar graphs with the data (Design your own Class)

- Modify bar_graph.py file to draw bar graphs.
- We are going to visualize 'data' list. The first item of the tuple will be labels for each bar, and the second item is indicating the length of the bar.



- Design your own 'Bar' Class to draw bar graph(bars and labels).
- If it seems confusing, follow these steps one by one:
    - Initialize class object and define a sample class instance. The sample could be the first item of 'data' list.
    - Try to draw a bar graph for this sample class instance. Draw rectangle by using class function. You can hardcode coordinates and size of the rectangle in this step.
    - Try to display labels next to the bar. Use class function as well. You can hardcode coordinates and font size in this step.
    - Instead of defining a sample class instance, try to create instances for all items in 'data' by using a loop. You'll need to use different coordinates and size for different items.
    - Get the width and height of the gaming screen. Use these variables to make your graph fit into the window. For example, if the width for bar section is 300 px, then we can draw rectangle for 'banana' to be 300 px since this is the largest value among our dataset. Likewise, 'apple' will take 100 px because the value of 'apple' is one third of 'banana'.
    - Change the size of the gaming window. Your graph should be able to fit into the new window.

Step 3 : (Bonus) Try another visualization

- Create another visualization for the same data. (eg. Bubble chart, Pie chart, etc.)