Week: 11 Date: 11/11/2021

15-110 Recitation Week 11

Reminders

- Check 6-1 due Monday 11/15 @ Noon EDT

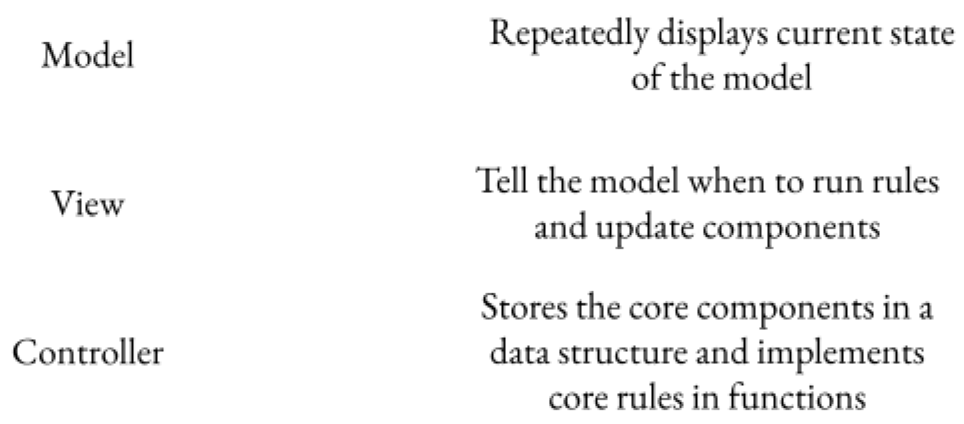
Overview

- MVC Review
- Simulations: Code Writing
- Code Writing with Modules

Problems

MVC Review + Simulation

Match the simulation part to its definition



Model	Repeatedly displays current state of the model
View	Tell the model when to run rules and update components
Controller	Stores the core components in a data structure and implements core rules in functions

Consider this setup from lecture:

```
    def makeModel(data):
        data["cx"] = 200
        data["cy"] = 200
        data["size"] = 50
        data["color"] = “red”
    def makeView(data, canvas):
        color = data["color"]
        canvas.create_oval(data["cx"] - data["size"], data["cy"] -
        data["size"], data["cx"] + data["size"], data["cy"] +
        data["size"], fill=color)
```

Write the function runRules(data,call) such that a new random circle (random x, y, size, and color) is drawn on the screen every 3 ms (don’t worry about the timer, that is done for you).

Debugging Large Projects

Reference the Tic-Tac-Toe code from a few lectures ago. There are 3 errors spread out in the code. Work with your classmates to debug this code and get the game working again!
Note: Remember some of the strategies that we have practiced for debugging! You can try adding print statements, talking through the logic with a classmate, and testing a variety of different inputs.

Bug 1:
Bug 2:
Bug 3:

Working with Data

Your Global Business professor gives you some sample corporate data to analyze more thoroughly for homework. Make sure to download the starter code and csv files from the 15110 website and store them in the same folder.
You will be implementing the following items:

1) First, we want to read in the corporate data from the csv file. Use the skills you learned in class to write the function **readCSVFile** that takes in a file path and saves the contents to a 2D list called data.
2) Write a function **departmentNameDict** that stores the information in an input list as a dictionary. The input list is a 2D list where each inner list contains 4 elements: department, department_name, location_id, and department_expenses. The output dictionary should be created so that for each inner list, there is a key of that department_name and an associated value which is the 2 element list of the corresponding lcation_id and department_expenses.
3) Write a function **departmentInfo** that takes in a dictionary like the one outputted above and a list of departments, and finds the mean expenses across the given departments and the most common location among the departments and prints them to the user. You may want to import a package to help you with this function