**Yazan Bawaqna**

**CS152 - B**

**Project 7**

**Graphic modeling for a ball collision using Zelle graphic in Python**

**Abstract:**

The goal of this project is to output a graphic visualization using Zelle graphic in python for a ball collision following the physics laws. First, the project simply offers the exploration of some Zelle graphic tools like “draw” for example. Then, we defined the objects that are going to be visualized and simulated. The process of doing the project involved the usage of “python classes”. A class in python is a command to construct an object, in this case, it is the “ball” and the “block”. Using classes, we can give attributes or in other words characteristics to the object, and here is where the power of classes appears. For instance, the characteristics of the ball include color, radius, velocity, and others. Furthermore, it is important to mention the role of the getters and setters which are extremely valuable in python coding. Those tools allow for assigning the attributed values. To explain, using a getter for the velocity attribute will give back the value of the current velocity, while a setter will reset the value to the new one. From here, we can keep changing the velocity of the ball and demonstrating a real-life simulation.

The project offers a wide usage of computational thinking commands, including “if statements” and “nested if statements”. The way they work is that they offer a condition for the program to run, for instance, if it is a collision between the ball and the block, then run this, else run this. Moreover, the use of the random library offers real-life simulations, as it gives different simulations for the same collision to account for unaccounted random errors.

**Solution to tasks:**

The most challenging part was the last task requiring recording a video of an interactive game or film. The way I approached this challenge is by using a main function. This will allow my function to be run all the time using if name == main and running the terminal. Next, since I am using Zelle graphics, I knew that I need a window to visualize my graphics. Simply after I defined my objects “the ball” and “the block” by connecting them to the ball in the physical objects file.

My next step which took me time to figure was to set the attributes. I have made the mistake of confusing the get and set functions due to their complementary functions. However, eventually, Ellen helped me correct the mistake and I changed the function into a setter function.

The difficult part came next, I used checkMouse() and getMouse() to control the object using the letters “a” and “d”. If the user clicked on “a” for instance, the set position function will be called which will reset the position of the ball by decreasing it vertically by 2 points. The issue I faced with this method is that I can lose the ball outside the window if the user kept clicking upward or downward. For this case, I used an if statement to reset the position of the ball to the center of the window if the ball got outside.

Lastly and most importantly, if the ball is in a collision, I will invert the speed of the ball but then adjust the direction with a random variable to keep the ball moving around in a random manner.

**Results:**

Outputs 1, 2 and 3 are within the videos.

Two videos are attached along with the report within project07, please check the folder

**Reflection:**

What was really fascinating about the project is how easy is it to level up the complexity of the project by importing outside-sourced libraries. Although ignorant as I may seem. I never comprehended the volume of outside libraries we can import into our code. While I was doing the project, I went to check for some online libraries out of curiosity. To my surprise, I came to know that there are endless options of codes to be reused at all levels of complexity. The power of this is that it allows beginners like me to have the ability to invent advanced projects, like sensors or 3D graphing. What’s most beneficial is that code reuse here allowed me to decrease my code lines by at least 75%, which is what is needed for a code to be efficient. While many think that reusing the code takes on the advantage of learning the actual code, however, this is misleading. From my experience, when I was reusing the Zelle library, I needed to teach myself far more concepts than I would otherwise have learned. That made me research getters setters, windows in python, and even more libraries. Reusing the code made me watch videos to use those libraries, perhaps that’s how I come to learn some trinker graphics as well.

**Follow up questions:**

***1. What is the purpose of having to get and set functions for a class? Why not just access the***

***object fields directly?***

The use of get and set function is to encapsulate the data, which helps avoid direct access of the data.

***2. Given a list of Zelle Graphics objects, write a for loop that would move each object in the***

***list by (5, 10) in x and y.***

List = [Zelle objects]

for item in self.vis:

            def setPosition(self, 5, 10):

#assign to x\_old the current x position

x\_old = self.position[0]

# assign to y\_old the current y position

y\_old = self.position[1]

# assign to the x coordinate in self.pos the new x coordinate

self.position[0] = px

# assign to the y coordinate in self.pos the new y coordinate

self.position[1] = py

# assign to dx the change in the x position times self.scale

dx = (px - x\_old)\*self.scale

# assign to dy the change in the y position times -self.scale

dy = (py - y\_old)\*(-1)\*(self.scale)

# for each item in the vis field of self

for item in self.vis:

# call the move method of the item, passing in dx and dy

item.move(dx,dy)

***3. The Zelle GraphWin class has functions getMouse() and checkMouse(). What is the***

***difference between them?***

getMouse(): waits for the user to clicks the mouse and returns where the mouse was clicked as a point

checkmouse(): It returns the last point on the screen where the mouse was clicked as a point. Which means it does not pause the function If the mouse is not clicked after the first time.

***4. What is the difference between simulation space and visualization space?***

Simulation is simply the production of the data as an output, Visualization however is the representation of those data, which can be in the form of graphs or graphics

**Reference:**

Ellen Lie, Colby’24, worked with me on my interactive game code

Kalyan, Colby’26, Debugging my code

David Roberts, Contributed In Issues regarding the video

<https://www.visualcomponents.com/resources/blog/simulation-vs-visualization-difference/#:~:text=In%20short%2C%20we%20define%20simulation,as%20if%20they%20were%20real>.