Project 2: Ping-Pong Simulation

- Project starts on: **Wednesday, September 30th**
- Project is due on: **Sunday, October 18th, 23:59**
- This is an individual project

Please read the entire handout before you start coding

Introduction

In Lab05, you created a cool animation which moved the square Bob. In Project2, you are going to build on the top of Lab05 and create a simple ping-pong paddle which deflects a ball.

In this Project, you are first going to simulate a remote controlled robot using two Python graphics windows. The first (named ‘Floor’) will be 400 x 400 pixels and will contain a 40 x 40 pixel black square drawn in the middle. The black square represents the robot to be controlled. The second graphics window (named ‘Control’) will be a 200 x 150 pixels with four blue boxes. The black square represents inside the Floor window represents the robot (named ‘Bob’). The user should be able to move Bob around the Floor by clicking on the blue squares in the Control graphics window. You need to make sure that Bob will not move past the boundaries of the Floor window. In this Project you will be adding a bouncing ball which bounces when it hits the Floor walls or Bob. That is when the ball hits the boundary of the Floor or hits Bob, it should bounce off in the opposite direction.

Your final Project should look like this.

`project_2_demo.mp4`

Setting up the environment

Go to your working directory in “data.cs.purdue.edu” and create a directory “cs177/project2”. Refer to the lab1 or project1 if you need to remember the steps to do so.

Project skeleton

Your project skeleton is here. Download this file and start working on it! Points will be deducted if you don’t adhere with the skeleton
Part 1: Create Two Graphic Windows

Part 1 is similar to Lab05. You are going to create 2 graphic windows. The first (named ‘Floor’) will be 400 x 400 pixels and will have a 40 x 40 pixel black square drawn in the middle. The black square represents the robot to be controlled. The second graphics window (named ‘Control’) will be a 200 x 150 pixels with four blue boxes arranged as you see in the drawing below. The black square represents the robot (named ‘Bob’) which the user should be able to move around the Floor by clicking on the blue squares in the Control graphics window.

The following steps will help you to realize Part 1:

1. Initialization: Create two graphic windows, the Floor and the Control windows, make sure to configure the windows as described earlier.
2. ControlPanel: Write a function named ControlPanel that does not accept or return any parameters. The purpose of this function is to draw the buttons in the Control window to represent the movements “Forward”, “Back”, “Left”, & “Right”.
3. checkButton: Write a function named checkButton that accepts no parameters but returns two integers separated by a comma. The function should check for a mouse click in either the Floor or the Control graphics window. If a mouse click is detected in the Control window, the function should determine if it was on one of the four blue buttons. If so, the function should return the appropriate values that will represent a dx and dy movement of Bob the robot in the Floor window. Bob will always move in increments of 20 pixels in any given direction.

You can refer Lab05 if you need more details for Part 1

Part 2: Create the bouncing ball

The following description will help you to create bouncing ball:

Write a function named makeCircles that creates a list containing a single entry; a red filled circle (bouncing ball). The radius of the circle must be 8. The starting (x, y) location of the bouncing ball should be random values between 100–300. Now add two randomly selected integers to the list. They should be either 5 or -5. These integers represent the amount by which the bouncing ball moves. At the end, the function should return the list.
Part 3: The bounce function

Now we will write the bounce function. This function is written to check if the bouncing ball has reached the boundary of the wall. If so, then we need to reverse the direction of its movement. For this:

Write a function that accepts two integers as parameters. If the first integer is either less than 8 or greater than 392, the function should return the inverse value of the 2nd integer, (that is: multiplying it by -1). Otherwise, the function should return the 2nd integer unmodified.

Part 4: The main function

Now we will define the main function. Inside the main function, do the following:

1. ItemCall ControlPanel to draw the controls.
2. Draw a black, 40×40 rectangle named “Bob” in the center of the Floor window.
3. Using makeCircles function, create the bouncing ball.
4. Ordered List ItemCreate a while loop that will continue until a click is detected in the Floor window. Inside your while loop do the following:
   ○ check if there is a collision between the ball and the rectangle Bob. If there is a collision, then the ball should bounce off in the opposite direction
   ○ check to see if the ball is hitting the boundary of the floor window; If so, then the ball should bounce off the walls
   ○ Call checkButton() to see if there is a mouse click in the controls panel and move Bob accordingly. Make sure that Bob stays inside the boundary of the Floor window.
5. Close the Floor window.
6. Close the Control window

Turnin Instructions

Run putty and login to data.cs.purdue.edu. Turn in your project by typing:

```
data 51 $ cd cs177
data 51 $ turnin -v -c cs177=COMMON -p project2 project2
```  

Grading Rubric

This project is worth 100 points.

<table>
<thead>
<tr>
<th>TODO</th>
<th>MAX POINTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part 1</td>
<td>10</td>
</tr>
<tr>
<td>Part 2</td>
<td>20</td>
</tr>
<tr>
<td>Part 3</td>
<td>20</td>
</tr>
<tr>
<td>Part 4</td>
<td>50</td>
</tr>
</tbody>
</table>

Computer Science Courses - http://courses.cs.purdue.edu/