**Lab 1 – Simple Obstacle Course (Student Manual)**

**Pre-Lab:**

* Have Unity and Visual Studio Code installed on your computer
* Have Visual Studio Code properly linked to Unity
  + For reference: <https://code.visualstudio.com/docs/other/unity>
* Watch this video to get acquainted with Unity: <https://youtu.be/IlKaB1etrik>
* Imagine a robot moving in a tunnel with some obstacles in its path that it needs to navigate around. The obstacles can leave either the left or right side open (or neither!). To navigate past the obstacles, whenever it hits an obstacle, the robot needs to determine whether it needs to move to the right or left to get past the obstacle, or to stop if neither option is available. Write the pseudo-code for the robot’s movement in the tunnel. It’s OK to leave this pseudo-code fairly high-level and informal, but it should touch on the broader issues that need to be tackled for this navigation task.

**Lab:**

Part 1:

* Follow your TA’s tutorial to get acquainted to the Unity interface and some basic terminology
* TODO: Finish designing a simple block moving continuously in a straight line on a flat platform and show it to your TA when you are done
* BONUS: figure out how to get the camera to follow the player object
  + Hint: look at the components given to you

Part 2:

* TODO: Implement the necessary functions in the P2PlayerMovement script for the player object to navigate through the obstacle course
  + Hint: think back to the pseudo-code logic from the pre-lab if you are stuck
* BONUS: Implement the ChangeColor() function to change the player object’s color based on the color input
  + E.g. calling ChangeColor(“red”) should change the box to red
  + If you can get it to work, try inserting it in different functions in the P2PlayerMovement script and see how the color reflects the logic in your code

**Post-Lab:**

Submit your P1PlayerMovement and P2PlayerMovement files as well as the answers for the following questions on Canvas:

1. What type of logic was used to determine whether the player object should dodge?
2. Let’s think about this simulation using some of the tools from physical symbol systems. What symbols does this machine use to represent its own movements (think about our *boolean* variables)? What do these symbols designate in the simulated world?
3. Which part of this lab did you find most challenging?
4. What do you think could be improved for this lab? (not for marks, but feedback is greatly appreciated!)