**Intro to Python**

**Exercise : Dodgeball**

**Challenge**

Modify the game so that objects can come from the other sides of the board.

**Suggestion**

This will require several changes. We will first need to make the `DodgeballBall` class support different direction movement. Then, we’ll need to modify the `DodgeballData` class to use it correctly.

The steps we are looking at here will allow the ball to move from left to right or from right to left. With a little bit of creativity, you should be able to also make the ball move from top to bottom and from bottom to top.

**DodgeballBall Changes**

First, in `DodgeballBall`, you need to add a data member that explains which direction the ball is traveling. We choose to use `self.direction` with a value of 1 meaning left to right and 2 meaning right to left.

In `__init__` add a new parameter `direction` to the end of the parameter list, and use it to set the data member `self.direction`.

Add another method `setDirection` that receives `direction` as a parameter, and sets the data member `self.direction`.

Next, in `DodgeballBall` add a `moveLeft` method that will move the ball to the left by subtracting from `self.x` instead of adding to it.

Now, add a `move` method to `DodgeballBall`. This method will use `if` and `elif` to check the value of `self.direction`. If the direction is left to right, it will call `self.moveRight()`. If the direction is right to left, it will call `self.moveLeft()`.

Finally, add to the `checkPosition` method so that if `self.x` is less than 0, the ball will be marked as not alive. Be sure to leave the check for `self.x` being too large.

**DodgeballData Changes**

These changes will make `DodgeballData` use the new features of `DodgeballBall`.

First, in the `newGame` method, you need to add the default direction of travel when the `DodgeballBall` is created. Do this by adding another parameter to the end of the parameters. Remember 1 means left to right and 2 means right to left. Choose left to right.

Next, we need to change the way the ball is moved. In the `evolve` method, change the call to `moveRight` to `move`. This way, the ball will know which way it is traveling, and can move correctly.

Finally, we need to change the way the ball’s `reset` is done. We choose to flip a coin to choose which direction the ball should travel. We then need to update the direction of travel for the ball, and update it’s position to be on the correct side of the field. We ended up with code like this:

```python
coin = random.choice( [1, 2] )
if coin == 1:
    self.ball.setDirection(1)
    self.ball.reset(0, self.height/2)
else:
    self.ball.setDirection(2)
    self.ball.reset(self.width-1, self.height/2)
```

**Up/Down Motion**

Can you adapt the changes above to allow for 1 of 4 options?