Computer Science 220L
Laboratory 9 – Tic-tac-toe

Learning objectives:

- Use Python's built-in list functions
- Practice constructs developed until this point in the semester through a game of tic-tac-toe

Part 1. Writing functions and `while` loops – with a partner

a. Write a function `calculateSum(value, numIterations)` that accepts a float, `value`, and an integer, `numIterations`. This function should add `value` to itself `numIterations` of times. Make sure to use a `while` loop. (E.g., for the call `calculateSum(3, 5)` the function should calculate and return 3 + 3 + 3 + 3 + 3 or 15. Assume that the multiplication operator does not work.)

Add code to `main()` to test `calculateSum()`.

b. Write a function `areEqual(num1, num2)` that accepts two float values and returns a Boolean value based on their equality.

Add code to `main()` to:

1. `Call calculateSum(.1, 10)` storing the answer into the variable `result`.
2. Execute `areEqual(1.0, result)` and stores the answer into a variable called `equals`. Should the answer here be True or False?
3. **Print "The two numbers are equal" or "The two numbers are NOT equal" based on the value of `equals`.**

Part 2: Build a tic-tac-toe game - alone

For this assignment you are asked to write a modular solution to the game of tic-tac-toe. Here are functions you will need. For each function add code to test them in main(). I have written the list below in an order that I think functions should be developed. As desired, you may have additional functions.

- A method to build the board. This method should create a list of the numbers 1 – 9 and return that list.
- A void method to display the board. (See sample displays below.)
- A void method to fill a spot on the board. This method will need to have the board, the position to be filled and the character to place in that position. Do some error checking here so that the board doesn't allow for letters other than 'x' and 'y'.
- A Boolean method to determine if a spot is a legal spot on the board. Don't allow the method mentioned in the previous bullet to execute if the spot isn't legit.
- A Boolean method to determine if the game has been won. (This requires a little thought. You can always skip it by just having it return False and thinking about it later.)
- A Boolean method to determine if the game is over. This should call the previously mentioned method plus check to make sure there are more plays allowed on the board.
- A method to play the game. This method should continue as long as the game is not over. Once the game is over, display the message "Player 1 wins!", "Player 2 wins", or "Tie" as is appropriate.

Below are sample displayed boards.

The board original board before any players have played:

```
1 | 2 | 3
---------
4 | 5 | 6
---------
7 | 8 | 9
```

The board after player 1 played an “x” in position 5, player 2 played an “o” in position 7 and player 1 played and “x” in position 1:

```
X | 2 | 3
---------
4 | X | 6
---------
O | 8 | 9
```

Submission:

Upload the file (don't forget to submit).