Milestone Description

I have created two Python files: game.py and milestone.py. The file game.py imports the class Board with all its methods from milestone.py, serving as the user's interface.  The current program receives inputs from the user for row and column numbers, the number the marks need to consecutively appear to win the game and create a board to host the game. The game allows multiple rounds now.

I have also made a few changes to the class Board in milestone.py:

* **game.py:**The file imports all methods of the class Board from milestone.py and serves as the interface for users to play the game. I added one while-loop to the file that asks if the users want to start a new round of games (after winning or losing).
* **def\_init\_(self):** I changed the def\_init\_(self) function to allow the user to input the board's width, height, and the number the mark needs to appear to win the game consecutively.
* **aiMove(self, ox):** I have also changed the aiMove(self, ox) function in starter.py to aiMove\_3(self, ox) and create another aiMove(self, ox) function. The reason for this is that it is possible for the grid not to be three times three, so more loops and considerations are needed. In the function, if self.width and self.height are equal to three, it calls the aiMove\_3(ox) function. If it is not a three times three function, it continues the if-elif-else statement.

·

Thoughts on the program

The primary function I will work on is aiMove(self, move). The project's main theme is to determine the "best" position to take when there is an M times N board with an input of (x, y) from the user.  It gets more complicated as row and column numbers increase. My thoughts are generalizing some frequently seen models using existing functions(inarow\_Neast(), inarow\_Nsouth(), inarow\_Nnortheast(), inarow\_Nsoutheast(), etc.). If the game needs the marks to appear T times consecutively in a row, column, or diagonal, some classic models that will be implemented are:

* T-2 marks in a row, column, and diagonals with both sides empty
* T-1 marks in a row, column, and diagonals with a space on the side and another mark after the space (|X|X|X| |X| becomes (|X|X|X|O|X|).