**Project Description**

*‘Be an Emerging Leader’ create-your-own adventure leadership assessment*

Going through various assessments with them based on Organizational Behavior (OB) theory (70-311) as well as a fun game of Othello, “be an Emerging Leader” will hopefully inspire users to learn something about OB and about themselves while having fun.

**Competitive Analysis**

There doesn’t seem to be any activity similar to this from past projects, but the ‘Hogwarts’ and ‘Lights Out’ games come close in the idea that there are multiple different ‘mini-games’ that are throughout the general narrative. My project will be differentiated with its focus on the organizational behavior side of things.

**Structural Plan**

So I think I’ll organize the final project by several .py files – one ‘master’ file that’s delineated by different modes (code for the framework taken from the animation demo notes) and each mode to be a different stage in the entire narrative (i.e. home screen, game 1, game 2, scenario, results screens). The code some of the individual games that go in those modes will go in separate .py files that I will import in order to be called in the ‘master’ file.

* Home Screen
  + Landing Page (similar to that of Netflix’s landing page)
* Balloon Game (Balloon Analog Risk Test – behavioral test of risk assessment)
* Implementing game of Othello with game AI
* A personality test
  + One of which to be Conflict Resolution Style Assessment
    - From “Conflict and Negotiation Processes in Organization,” Thomas, K.W. (1992)
  + Another of which to be True Colors Personality Assessment
    - From Lowry (1978)’s True Colors assessment, but modeled more closely from Carnegie Mellon Emerging Leaders’ True Colors framework

**Algorithmic Plan**

Othello will be the most algorithmically complex aspect of my project.

*Othello*

Starting out with drawing the board (implemented via a 2-D list, similar to how Tetris was implemented), drawing the chips, placing the chips, placing chips at legal moves, and alternating between ‘black’ and ‘white’ chips at first. Then, moving forward with implementing game AI (minimax) framework for the ‘white’ color (computer). Essentially the general algorithm behind each player is:

Human (black chips): checking for possible moves on the board (one that is empty and is adjacent to another piece – i.e. that it’s not isolated to the side of a board), and then checking for all legal moves from that (legal moves being looking up/down/left/right/NE/SE/SW/NW there’s a white chip such that if I keep going in that direction and hit a black chip, then the current position (event.x, event.y) is a legal move.

Computer (white chips): going through it’s current chips, checking for legal moves for each chip. Legal moves defined similarly to the human such that for each current chip, in either of the 8 directions, that if there’s a black chip, I keep going in that direction. When I hit an empty spot, it is a possible move. Then, I collect each of these possible moves and test them with the minimax algorithm with the computer’s objective to find moves that will **minimize** the number of black chips on the board.

**Version Control Plan**

Github!

**Module List**

Good old Tkinter and game AI

**TP 2 Updates**

Othello (also known as Reversi) is a two-player game each denoted as ‘black’ or ‘white’ chips. In a 8x8 grid-like board, each player takes turn moving, where one can only place a chip in a position adjacent to another chip. Particularly, one must do so in a way that a piece must ‘flank’ opposite color pieces so to ‘flip’ those colors. Game ends when either entire board is filled or if both players have no moves to play. Winner is determined by the player with the most of their color chips on the board remaining.

Algorithmically, I start out representing the board as a 2-D list where elements can be ‘None’, ‘white’, or ‘black.’ The board gets updated when chips are being placed and flipped. In the folder I have two files relevant to Othello (othello.py and WIP-othello.py) with the former being the ‘last working version’ (manual alternating between players) and the latter work-in-progress (currently not functional yet) version where I’m working on implementing game AI. I’m starting out with implementing just a random strategy for the AI. Once it works, I’ll work on a more complex AI strategy (i.e. minimax or minimax with alpha-beta pruning).

**TP 3 Updates**

Not really any drastic design updates – all the functional pieces (othello.py, trueColors.py, conflictResolutionStyles.py, balloonGame.py) are there, and put together through the main.py and mostly UI navigating through them, whether it’s mousePresses or keyPresses.

*Special note to TP Mentor (Harrison)*

It was really nice working with you, glad I got to know you through the project – no regrets taking this class/doing the project on my part. All the best with your future endeavors!