Project 4: Tic Tac Toe

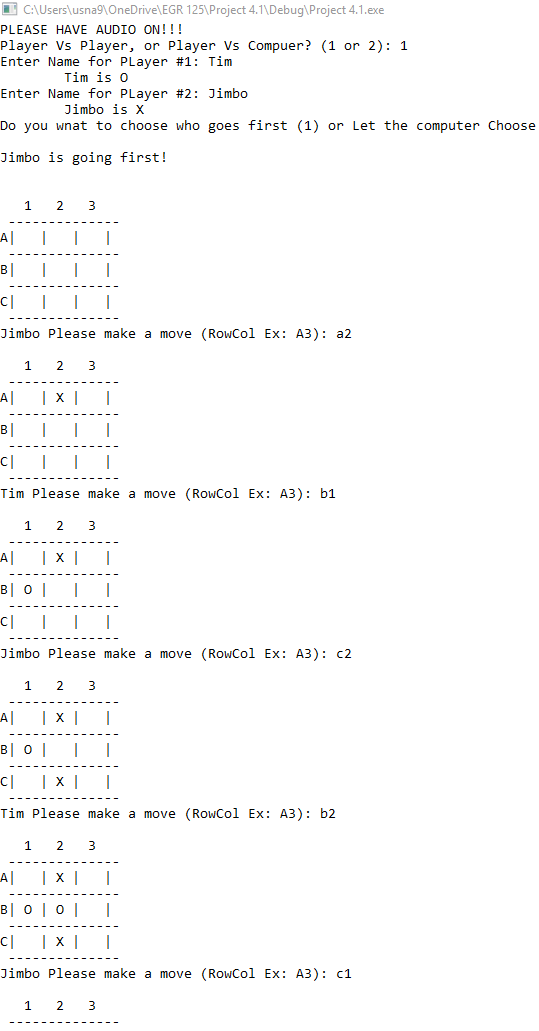
Student: Thomas Laverghetta

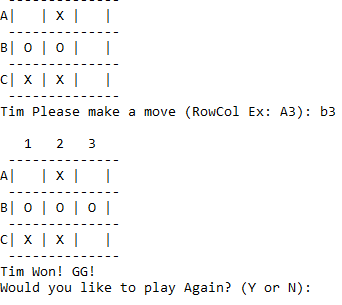
Course: EGR 125

Instructor: Prof. Grimes

Test Results

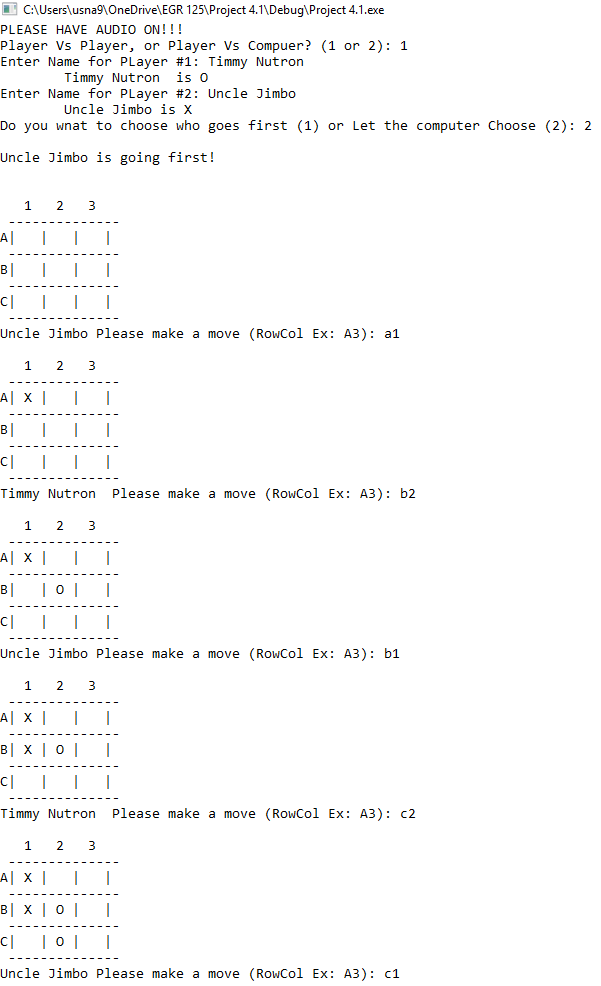
Case 1. Player 1 wins with a horizontal TicTacToe:

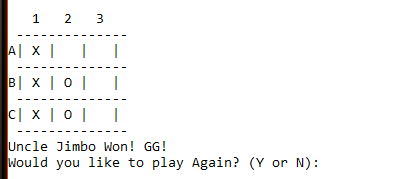




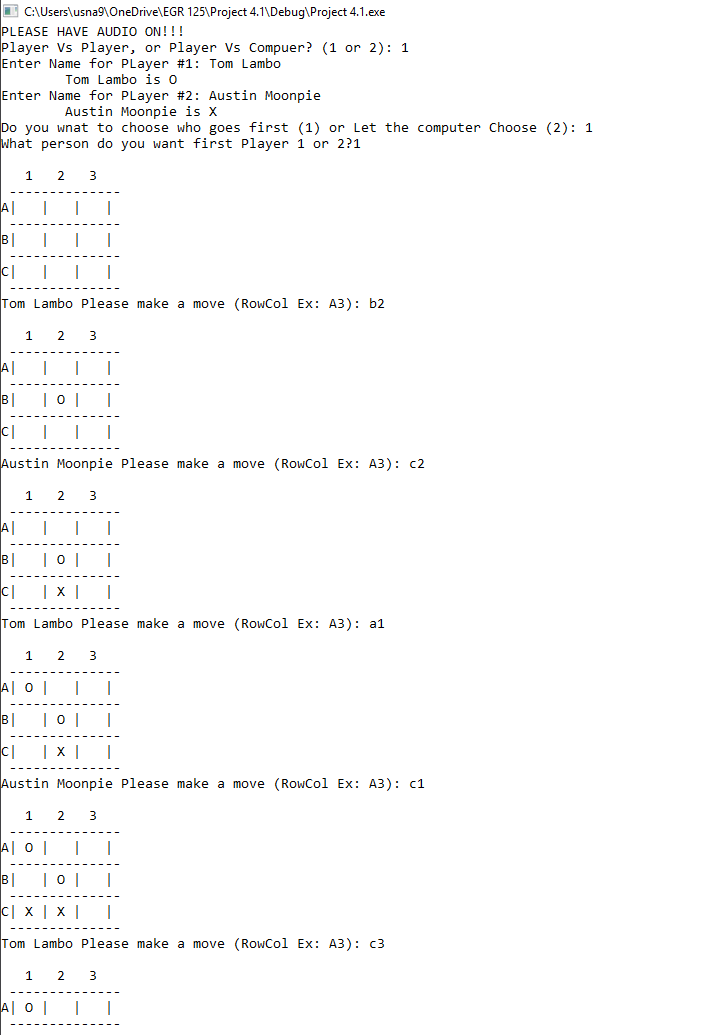
if kind a look funny because I had to take two screen shoots.

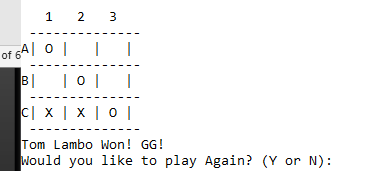
Case 2. Player 2 Win vertically





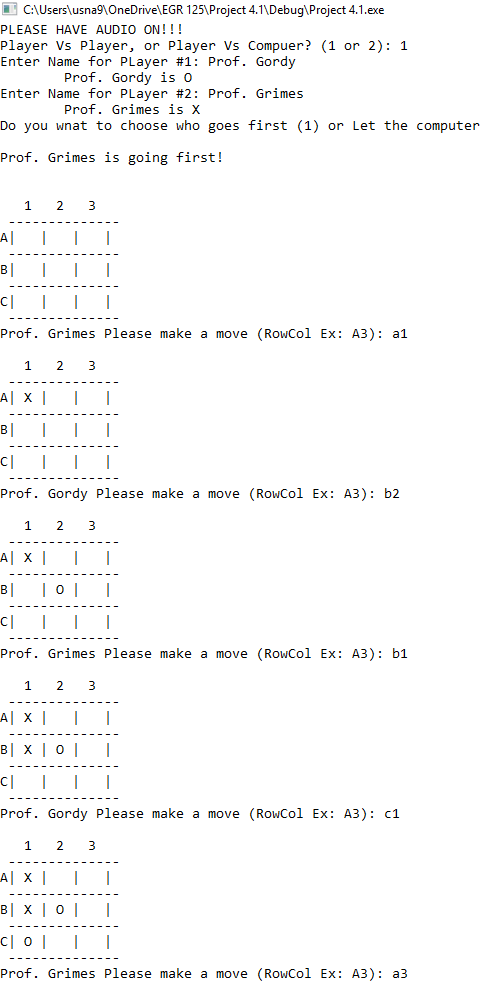
Case 3. Player 1 wins with a diagonal:

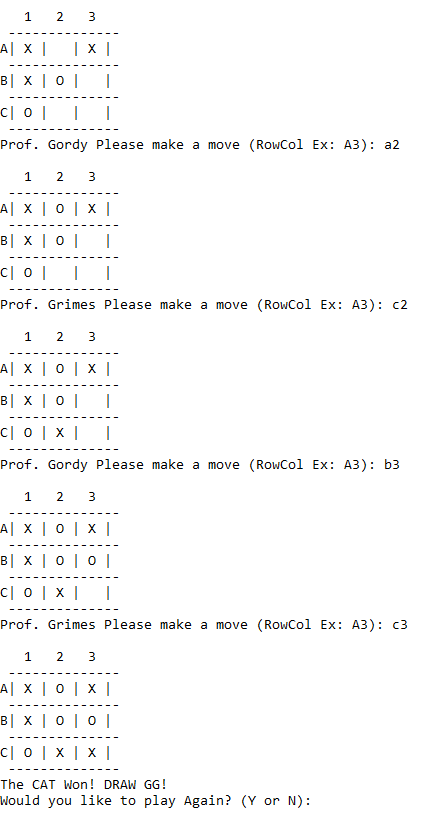




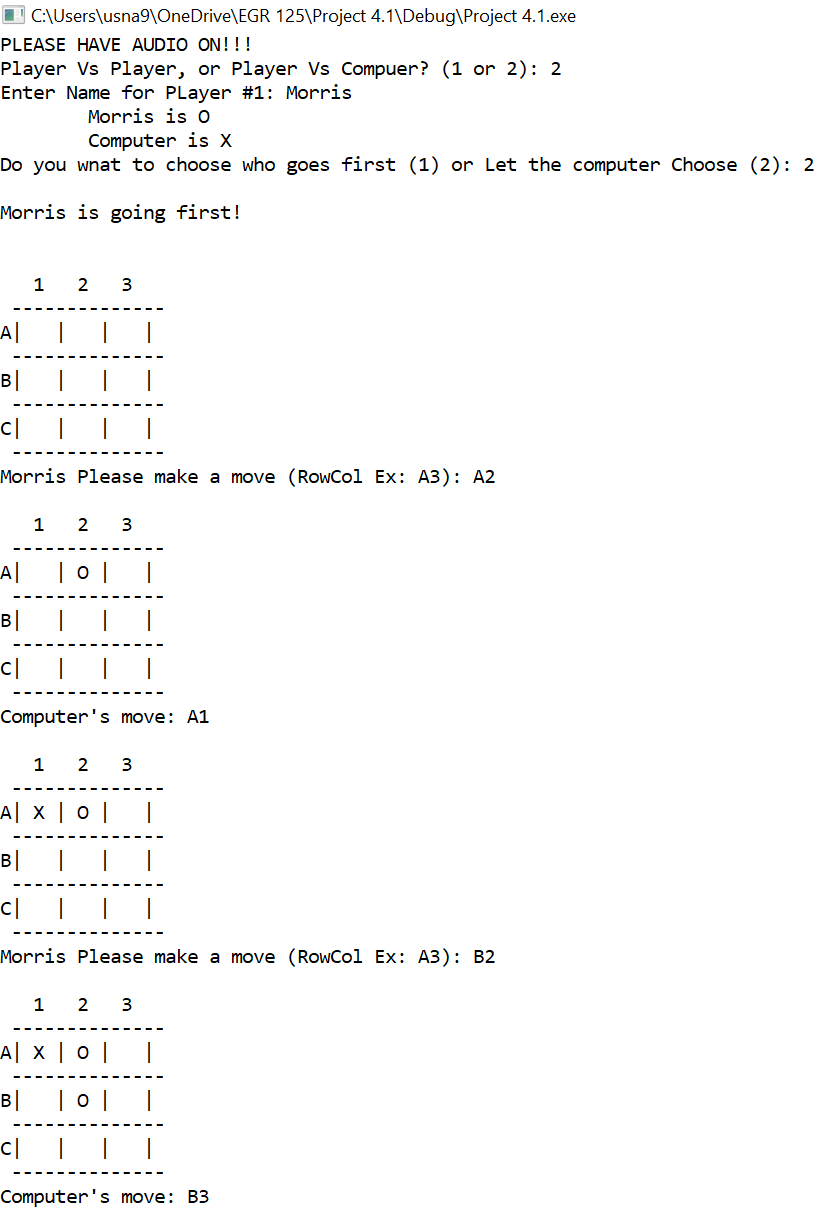
Two Screen shoots, that’s why it looks bad.

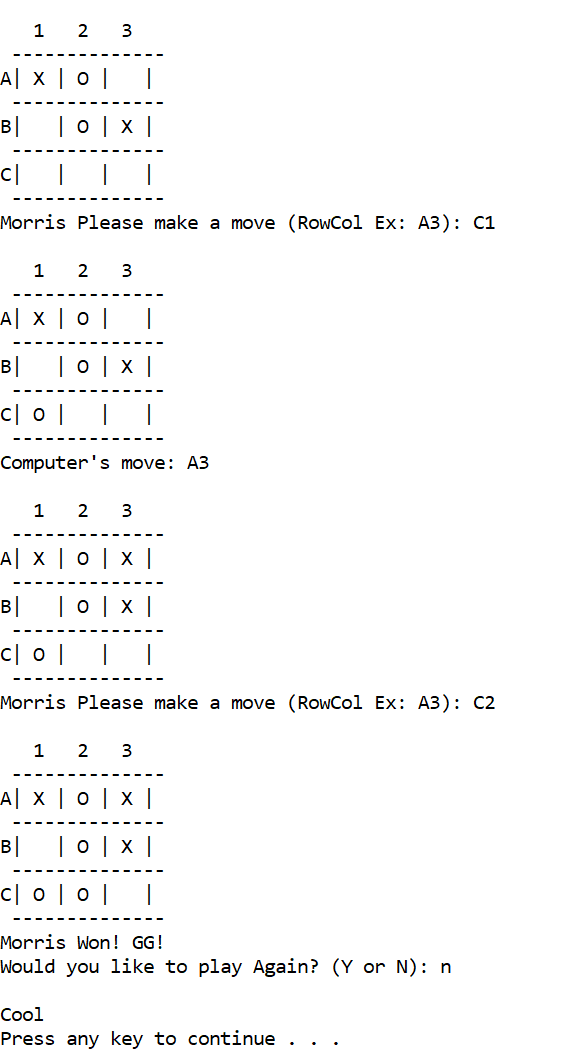
Case 4. CAT wins





EXTRA CREDIT Test, AI.





Class Diagram

|  |
| --- |
| **TicTacToe** |

|  |  |
| --- | --- |
| Data Fields | Descriptions |
| Size: static const int | Size of the board |
| rowMove, colMove: int | Variables to hold player moves |
| CBA: char | Hold player move in letter format |
| Player[2]: string | Hold players names |
| BoardMoves[Size][Size]: char | Board with players moves |
| Plays: map <string, char> | Takes players name (string) and assigns it a char value (char) |
| ABC: map <char, int> | Takes char (A through C) and converts into integer (0 through 2). For converting moves. |
| ReverseABS: map <int, char> | Does the opposite of ABC |
| +TicTacToe() | Default Constructor |
| +First\_Player(): int | Chooses Random player to become first |
| +Enter\_Player\_Name(int,bool): void | Player enters their name, and whether AI is playing |
| +Display\_Player\_Name(int): void | Display Player |
| +Make\_Move(int): void | Player makes a move |
| +Valid\_Move(): int | Validating player moves |
| +Display\_Board(): void | Displays board with moves made |
| +AI\_Moves(int&, int&): void | Computer Makes a move using RNG |
| +Game\_Over(int): int | Determining if the game is over (someone won) |

Discussion

Program Performance:

-The program performed to satisfaction. No problem with testing. Everything runs smoothly. Had friends break to make sure that it was debugged to satisfaction.

-Possible imitation: possible sound glitches when starting the program.

Extra Credit:

-I played music in the background while the player plays, and it changes as pending on actions taken by the user. In the beginning when the user chooses the type of game desired the program will play Jeopardy theme song. When the game starts (when the player play) it starts Pokémon battle music. Lastly, when the game is over, it will play music pending on who won. If a person wins, then it will play the victory song from Pokémon. If the computer wins, then it will play Knife Party’s 404 songs. If Cat wins, then it will play Nyan Cat.

-I made a program that allows the user to play against the computer (In the program I called it ‘AI’). The ‘AI’ is powered by random number generator, RNG. Even though in reality, the ‘AI’ Is really stupid, it still does the job, give the player another option if they don’t have anyone to play against.

Potential Improvements:

-Changing the music every time a player makes a move (sound effects). I would love it, if I had the time, to add sound effects for everytime a person plays a move.

-Maybe some more formatting

-Better AI. Instead of just RNG, maybe make an actual AI system.