Tic-Tac-Toe Project

For this assignment, I was working with Nick Becker. We were originally going to have three in the group but someone decided to work alone so we decided it would be best to split up the tasks amongst ourselves. This means during the architect phase we each had a hand in implementing ideas.

The design of our solution was aided by both Nick and I with some specific parts being encouraged which I will specify later. In Project 1's pdf file, there was a section of useful hints and we particularly liked Kylie Ying's tutorial so we wanted to use it as background to our own project. The first aspect of the design we went over was how the game was going to function at its basic level. For this we separated the project into two files: game_ and player_. I'll go over game first . The first step was the board. We wanted something in our design to be fundamentally different than Ying's to challenge us and differentiate the code. I came up with the idea to have the board be a dictionary where every square had a corresponding number. Nick was then able to figure out a method to have a working board and display board in the print. If you notice in the initial print we have a board that shows the corresponding number to each move but also the current board below that. The current board is then updated as the game is played. We then had to make the a move on the board. For this there were two aspects Nick and I had to make sure. For the player to select a move, the spot it picks has to be vacant otherwise the pick will be rendered invalid. Nick caught that when a move is made it should check if that is a winning move. This will not allow another move to be made because the game should already be over. Until writing the code, we remembered that it should also check which symbol was the one that won so we could put it in the final print statement. Checking for victory was easy because of the dictionary format. I told Nick to include vertical, horizontal, and diagonal but he likely would have included it anyways.

Last paragraph was on how we crafted the game, this will be on making the participants although I will separate the AI into its own. First, we knew we had to assign each player their own value (X or O) to know which player was who. Because we assign the value to the player, the next step was getting the character to make a move. Me and Nick had to breakdown at the basic level what makes a move. We agreed that the essential principles to make a move, as basic as it seems, is that the move has to be valid (meaning it has to be an integer between 1-9) and the move has to have not been made before. If these are true a move is allowed to be made. For the human participant they can make whichever valid move they desire. For the computer we needed to make him unbeatable because in TicTacToe if you know exactly how to play it is impossible to lose. Drawing is still possible though.

To explain the AI both files were involved. We knew for this assignment we were to use the minimax algorithm which for our game of TicTacToe, would simulate all possible game states to determine the optimal move. Our need was to figure out which parameters would allow it to come to the optimal move. To do this Nick had the idea of giving a value for a win (we later fleshed out how exactly to make the values with the help of the video). The AI would win if it got a higher score and lose with a lower so we made it it go through and bring back the highest score. A tie would just give 0 and this was a better outcome then losing as well because if both participants play well a tie is guaranteed. To be able to execute this fully there needed to be a couple items I made sure Nick had to include. We needed the AI to know if the game was terminal. We needed it to know that cells were vacant and how many were to get to the win as fast as possible. We needed it to keep track of the score. For the algorithm to work recursivley Nick noted to allow the algorithm to make a move, check the score, and revoke the move when needed. With that we had the architecture on how the AI would work.

Finally, we discussed how to actually start and play the game. We wanted to make a print statement describing the game and then a board that would show the human player which number indicated the proper square. There was a problem that Nick got which was that we couldn't have a board with the numbers get the players move so we decided to print another board and call it current board. When you first hit play this board was gonna be completely blank and as the player moved it would get updated. We knew that we needed to allow a player to make a move and after the made one the next move would be made by the other player. I made sure we had the board checked after every move to see if there was a winner or there were no moves left. Otherwise, no valid moves would remain and you would be stuck in a loop. Depending on the outcome a specified print statement would say that the person who made the last move won (if there was a winner) or there was a tie. To make the UI look nicer Nick made a statement that would say where the participant moved. In the end we decided to give X to the user and O to the AI and with that the idea for the code was mostly done.

Nick handled a majority of the coding phase but we had a lot of planning for it done already and we watched the video together so he was good with that. He made it polished and easy to go in and understand what each part of it does and the function. I handled the reporter role and just kept notes as we discussed and wrote made them more thorough for the paper. I also handled checking the AI for accuracy. In testing I played many games with it and did not win a single one. Something I found interesting is there is a way of winning in TicTacToe every time if you start in the corner. The only way to not lose if you go second is to go to the middle. Every single time I would pick a corner move the AI would go to the center so I knew our code worked. Overall, this assignement went pretty well, looks clean, and Nick was an excellent partner to work with. With more time, the biggest improvement we could have made is to move the game out of the terminal. For a previous class, I made a nice looking TicTacToe game so have the functions work there would have looked more appealing.

Reference to the Kylie Ying video: https://youtu.be/8ext9G7xspg