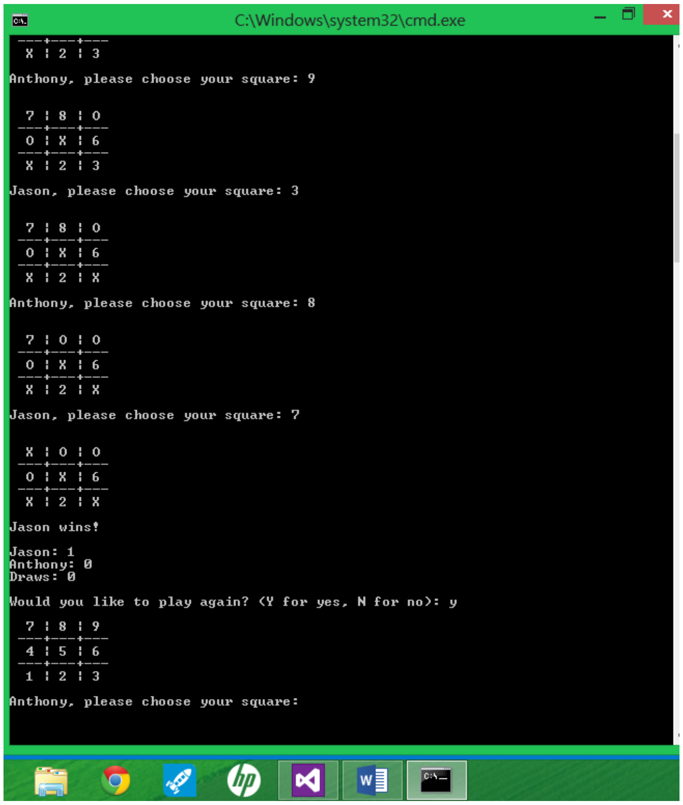
Jason Duffey
Dr. Estell
ECCS 1611
November 8, 2013
Tic – Tac – Toe

The purpose of this assignment was to design and code a functional tic-tac-toe program, that would allow players to choose what square they want to place their mark in, and the program would automatically determine when someone won, and who that person was. Creating this program has helped me to become more comfortable using arrays, functions, and putting my knowledge together into a real-world usable application.



```
 ---+---+---
  X ¦ 2 ¦ 3

Anthony, please choose your square: 9


  7 ¦ 8 ¦ O
 ---+---+---
  O ¦ X ¦ 6
 ---+---+---
  X ¦ 2 ¦ 3

Jason, please choose your square: 3


  7 ¦ 8 ¦ O
 ---+---+---
  O ¦ X ¦ 6
 ---+---+---
  X ¦ 2 ¦ X

Anthony, please choose your square: 8


  7 ¦ O ¦ O
 ---+---+---
  O ¦ X ¦ 6
 ---+---+---
  X ¦ 2 ¦ X

Jason, please choose your square: 7


  X ¦ O ¦ O
 ---+---+---
  O ¦ X ¦ 6
 ---+---+---
  X ¦ 2 ¦ X

Jason wins!

Jason: 1
Anthony: 0
Draws: 0

Would you like to play again? (Y for yes, N for no): y

  7 ¦ 8 ¦ 9
 ---+---+---
  4 ¦ 5 ¦ 6
 ---+---+---
  1 ¦ 2 ¦ 3

Anthony, please choose your square:
```

A Game in Action.

1. What are the key issues for implementing a Tic-Tac-Toe game with a computer program?

   There are many issues to consider when implementing a tic-tac-toe computer program. The first of which is displaying the game board, and the placing the player’s mark in the correct square. A second issue is determining if a player has three in a row. There are 8 possible ways a person can win in a game and the program has to be able to check if a player has achieved any of these 8 possible ways of winning after each turn. A third issue is resetting the board after each game, and having the player who went second for the last game go first in the next game, and alternating who goes first each game.

2. What data type and structure (i.e. array or vector) did you choose to represent your tic-tac-toe board with, and why?

   I chose a character array to represent the board in my program. I chose an array because the board is a fixed size and would not need to change size when the program is running. An array's size must be determined when you write the code, so an array is nice for this application since we know its size. I chose to assign the array's data type as characters because there are character literals for the numbers, and I can display the board with the unused squares as the numbers. In addition, once a player chooses a square I could easily change the spot in the array holding that squares information into the player's mark (either 'X' or 'O').

3. Describe the algorithm that you used to implement the hasThreeInRow function.

   To implement the hasThreeInRow function, I first had the computer check to see if the player had three of their marks in any row. I used a for loop construct in order to do this. It would check the three boxes in the first row to see if all of them were the player's mark, if not it would move on to the second row then the third row. Following that I had a similar for loop structure that would check the three columns. Finally, I had the program check to see if the player had three in a row on either of the diagonals.

4. How did you verify the correctness of the functions that you were required to implement?

   To verify the correctness of the functions that I implemented I created a second test project that I used to isolate the function by itself, or with the necessary information it needed to run and ran the test program to determine if it performed how it was supposed to. For example, the hasThreeInRow function had to be one of the last functions that I implemented because it required the getPlayerInput, isLegalMove, and placeMarkOnBoard functions in order to work. I played several games and had one of the players win in each of the 8 possible ways that a player could win in order to see if the function worked for all situations.