Name: Andre Freeman

Student ID: 410521320

Subject: Network Programming

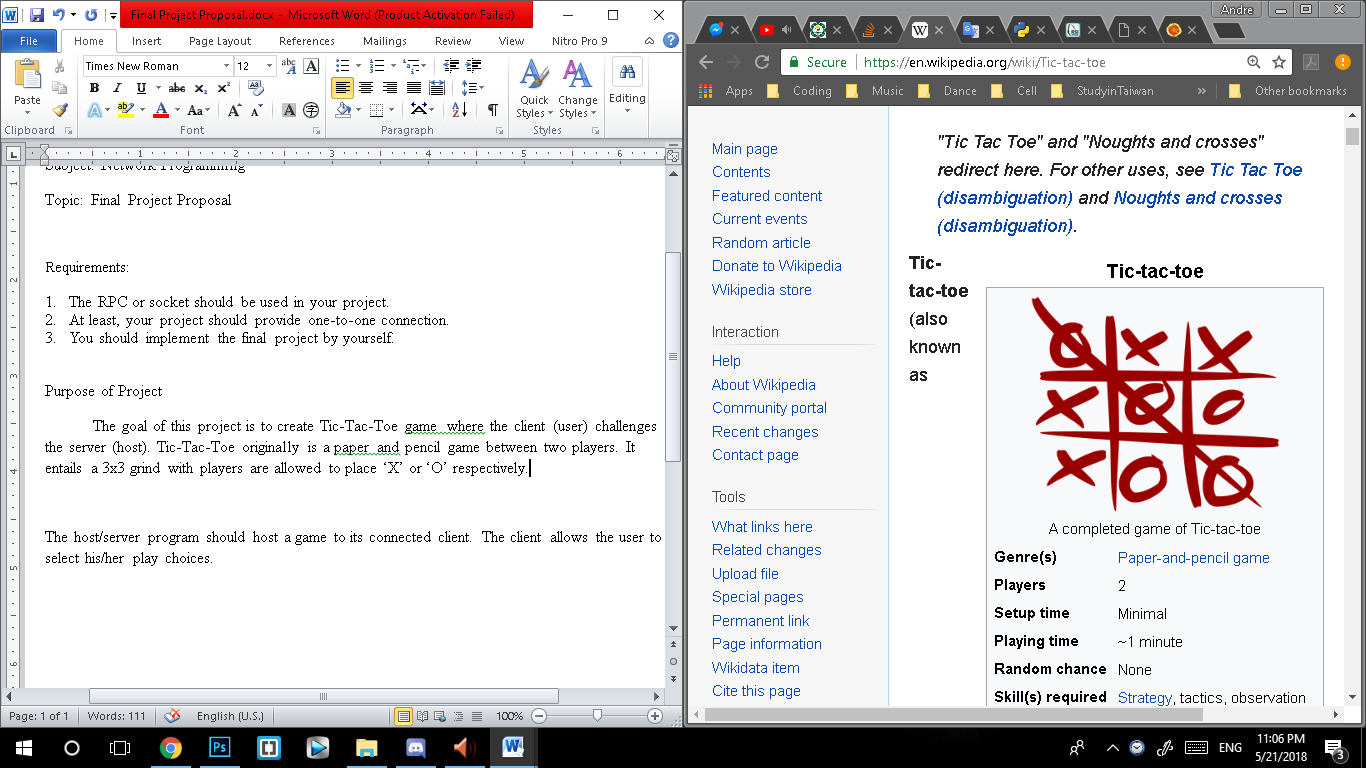
Topic: Final Project

**Requirements:**

1. The RPC or socket should be used in your project.
2. At least, your project should provide one-to-one connection.
3. You should implement the final project by yourself.

**Purpose of Project**

The goal of this project is to create Tic-Tac-Toe game where the client (user) challenges the server (host). Tic-Tac-Toe originally is a paper and pencil game between two players. It entails a 3x3 grind with players are allowed to place ‘X’ or ‘O’ respectively. The diagram below shows the pencil and paper layout of the game.



The host/server program should host and play the game versus its connected client. Between the server and the client they will place ‘X’ or ‘O’ respectively in efforts to gain three (3) consecutive ‘X’ or ‘O’ accordingly or until no more slots on the board is available. In the end indicate the results of the game.

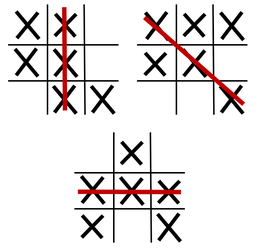
**Implementation**

The programs will be implemented using C++ by creating two(2) programs (i.e. the server-side program and the client-side program). The operating system (OS) of choice will be Ubuntu 16.04 x64. Also it is my desired wish that a learning algorithm be utilized in this project to predict/choose the host (server) playing choices. If not, then a simpler approach will be utilized. The client side should accept user choices and also display a representation of the 3x3 board for easy user interpretation. The choice would be transmitted to the server for evaluation. The server response will indicate the next possible and results to the client.

**Testing and Judging**

The client should be allowed to connect and submit the position of where he/she will like to play. The game should also indicate a winner for the specific conditions:

1. User or server is allowed to play 3 consecutive plays in a horizontal, vertical or diagonal manner (as shown below).



The server should make plays in efforts to win or prevent the user from winning. In the end the server should indicate to client “DRAW”, “I WIN” or “YOU WIN”. Also since this project requires the use of socket pairs the game should also be able to function across the network platform.

**Implementation**

To achieve one to one connection two programs must be create:

1. Server Side

2. Client Side

Also a standard must be put in place in terms of the data to be be transferred between the two programs, port number and IP address. For this Project we utilize 127.0.0.1, ‘localhost’ just for ease of trail and testing. The data that is transferable between both programs is a struct containing an integer value for a command set and received ‘cmd and the current Board state (i.e. the state of each position on the board.)

1. The Server side

The server side is responsible for the following actions.

i. init() function - Initialize the board by setting each position an integer value. Set the winner value to a no winner state of ‘-1’. Determine whether the server side is going to be playing as ‘X’ or as ‘O’.

ii. isChoice() function – Checks if the board has a playable position.

iii. PrepMessage() function – A fast and quick way to match the board state values to the message to be set to the client side.

iv. setWinner() function – Set the winner value to the corresponding winner. (Either CPU or client side User).

v. isWin() function – Checks the board states to see if there is a winner.

vi. isValid() function – Checks to see if the play is a valid or legal play.

vii. Aiplay() function – Calculates the server choice to be made on the board

This function also takes into consideration attempts to block the Client side user from winning. Possible win possibilities.

viii. run\_server() function – Initialize, listen and accept the socket structure and communication. Also is the main function that runs the game (i.e. call all the other functions)

2. The Client side

The client side is responsible for the following actions.

i. print() function - Prints of the board state received from the server side t o the screen to the user in a manner replication the actual game board layout.

ii. choice() function - Accepts a value from the user and checks if the value is in the constraints of the numeric position of the board.

iii. Main() function – Initialize and connect to the server through socketstream. Display to the screen the appropriate text message based on he received command ‘cmd’ from the server. Also accepts the user choices and send this value to the server.

N.B. Both the server and client side contain code to allow you to play again.

**Results**

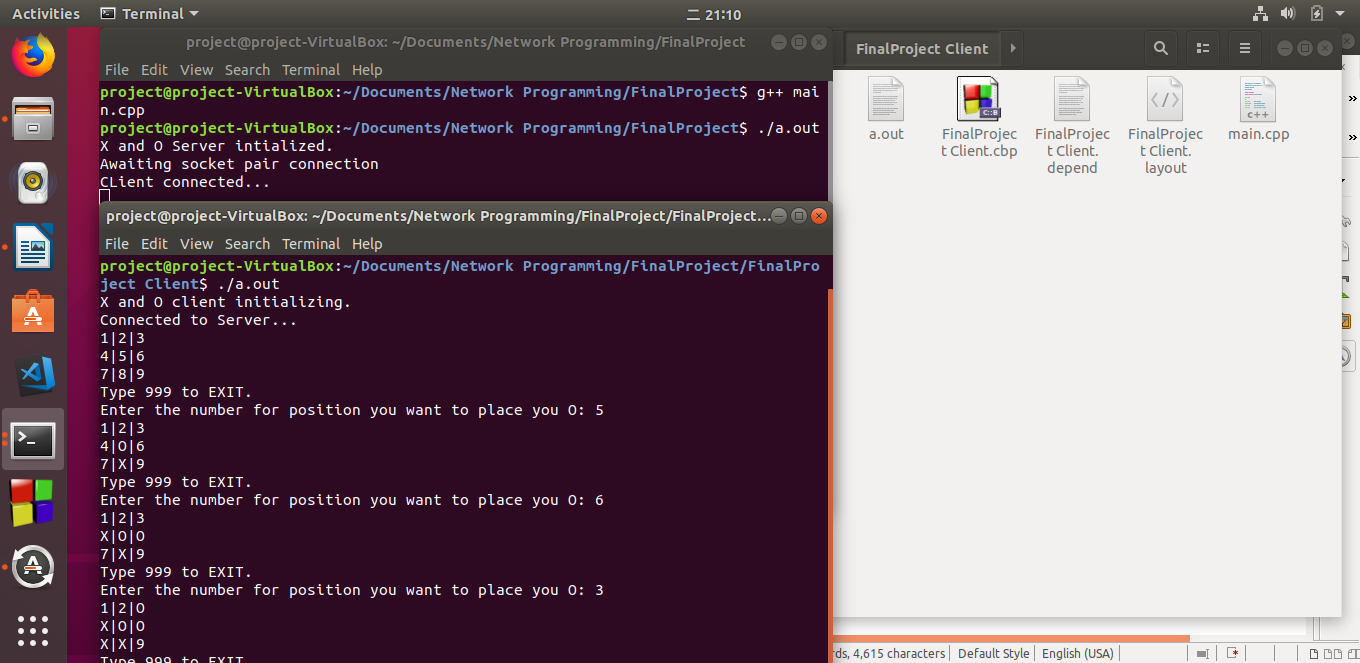
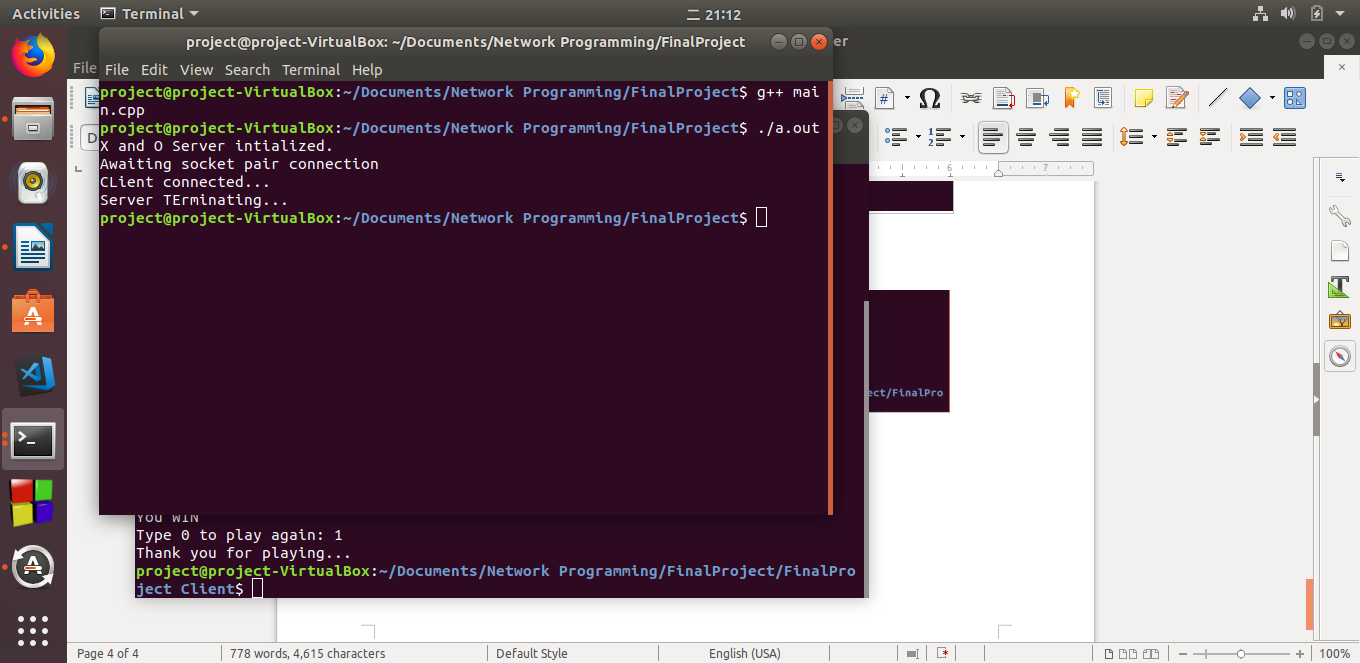
****

Figure 1.1 above showing the X and O game in progress.

****

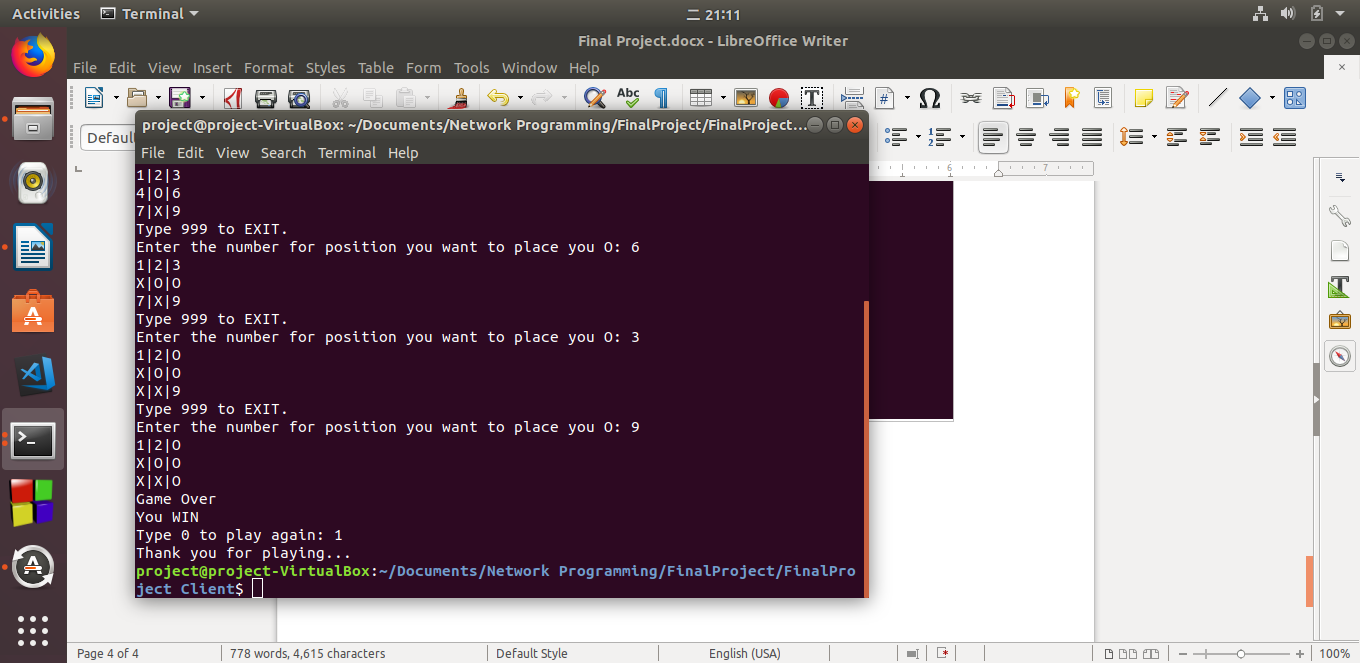
****

Figure 1.2 - showing the end result of the X and O game.