ECE114L – PROJECT 1

TIC TAC TOE

Kim Nguyen – Jim Son

November 18, 2017

1. **OBJECTIVE**

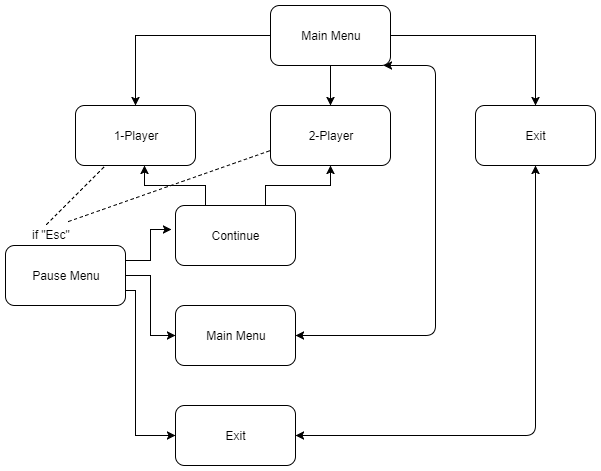
Build a C++ program for the game Tic Tac Toe, with 2 different modes: player vs. computer, and player vs. player.

1. **BACKGROUND**

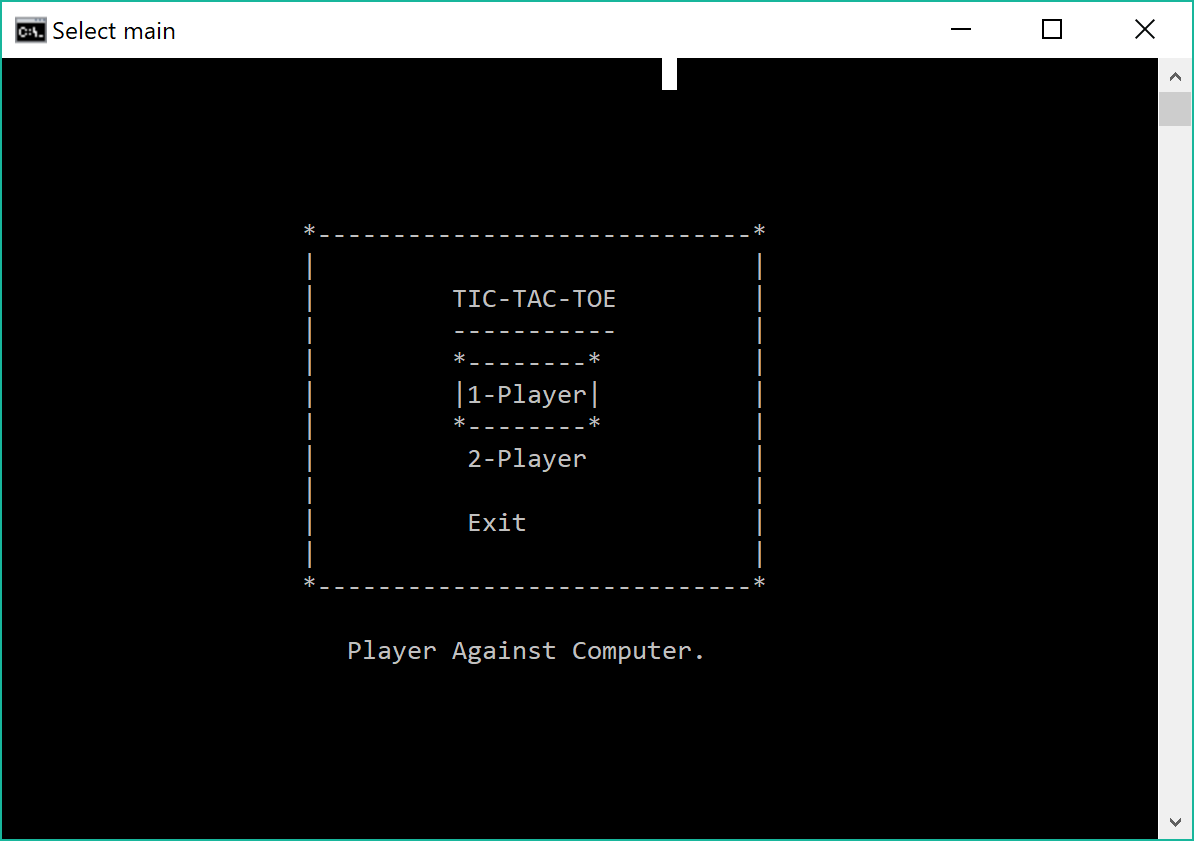
We had initially started this program using specific user inputs to select specific boxes on the playing field. Using this method brought many problems because it allowed the user to input characters that should not be inputted. We had run into an infinite loop when there was a wrong input. This was fixed, but other bugs remained. This idea eventually became scratched when further research was done on a user-friendly menu which was supposed to only be an addition to make the program more presentable after the game was finished. This menu used arrow keys to choose an option as well as the return/enter key to select. After understanding how the virtual keys were utilized for a menu, we realized it could be used on the actual game. This had ultimately made the program very user-friendly, and it also avoided the bugs from the previous idea of specific user inputs since the only available inputs are the arrow keys and return/enter.

1. **RESULTS**

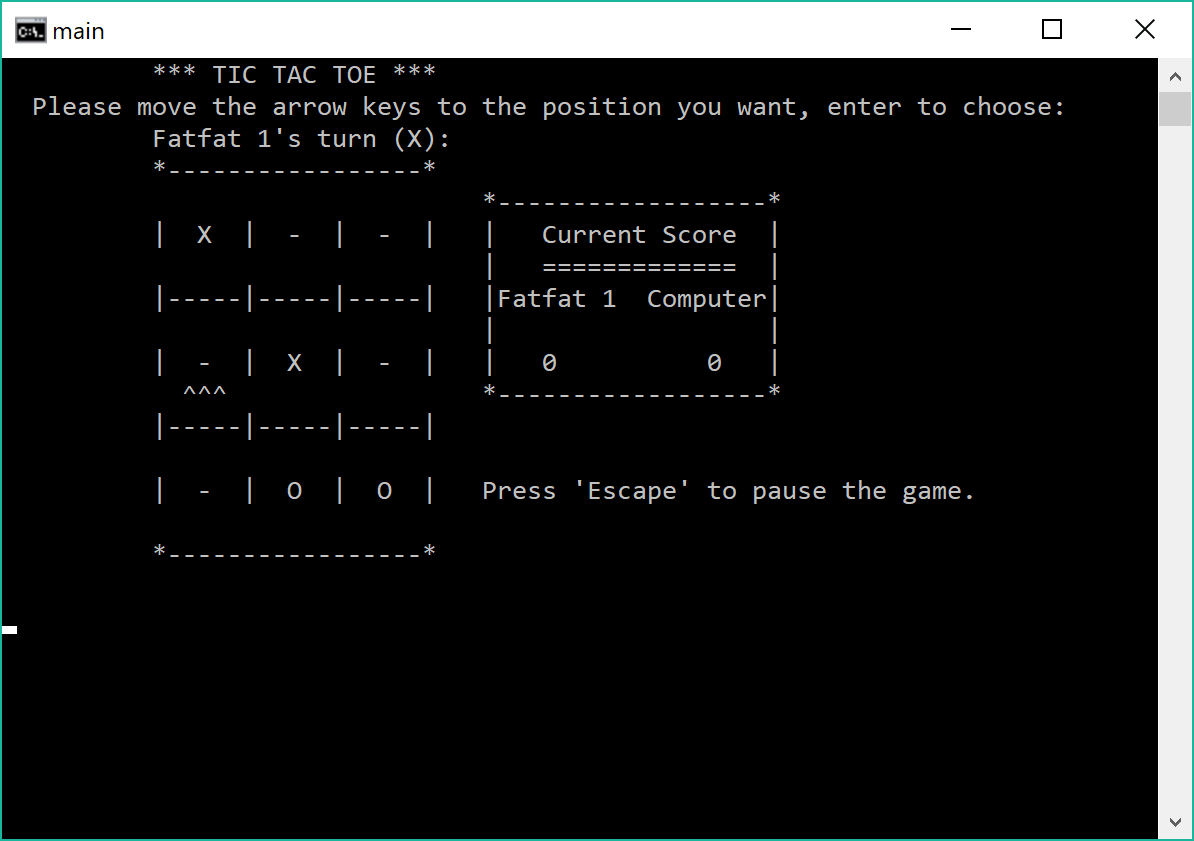
We created a program that asks user to use the arrow keys to move to and select the desired positions, and the Esc key to escape the program.

The steps of the program are illustrated in Figure 3.1:

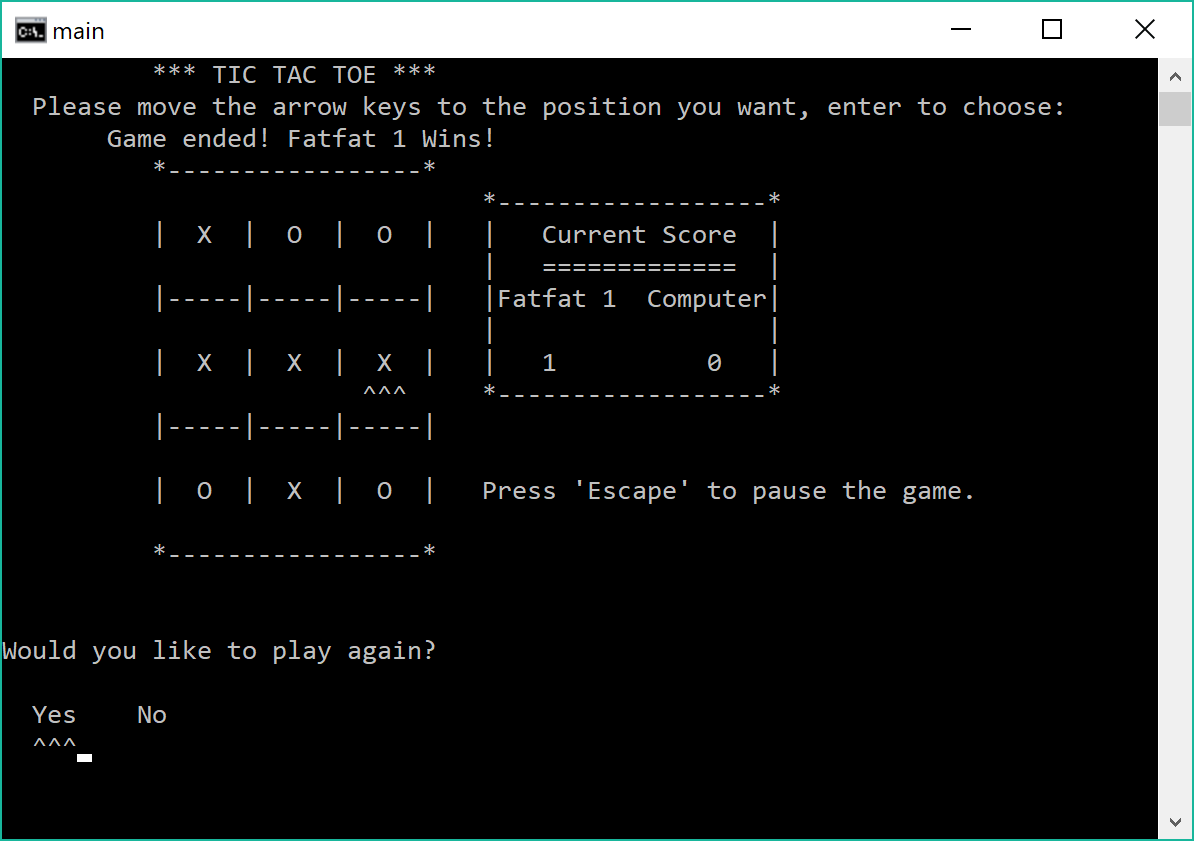
*Figure 3.1 – Program Flow Chart*



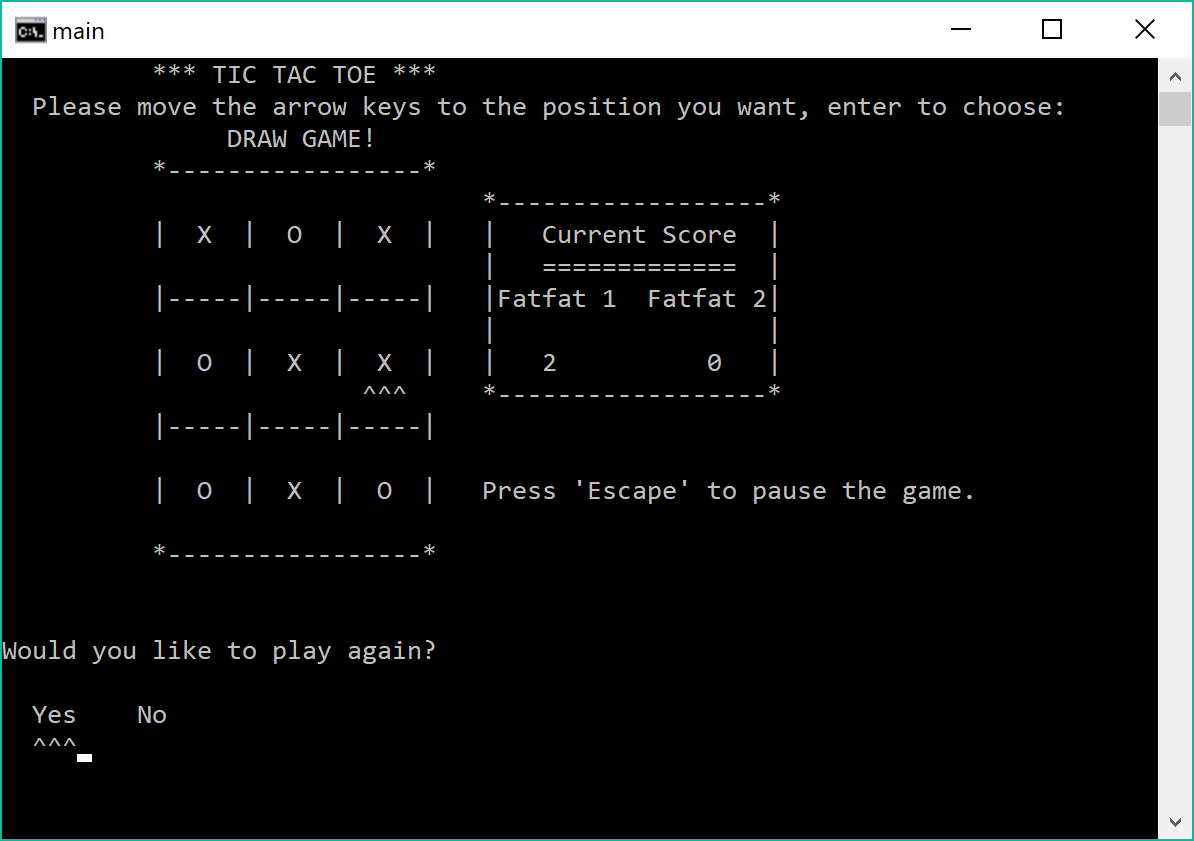
*Figure 3.2 – Screenshot of the program output – Main menu*



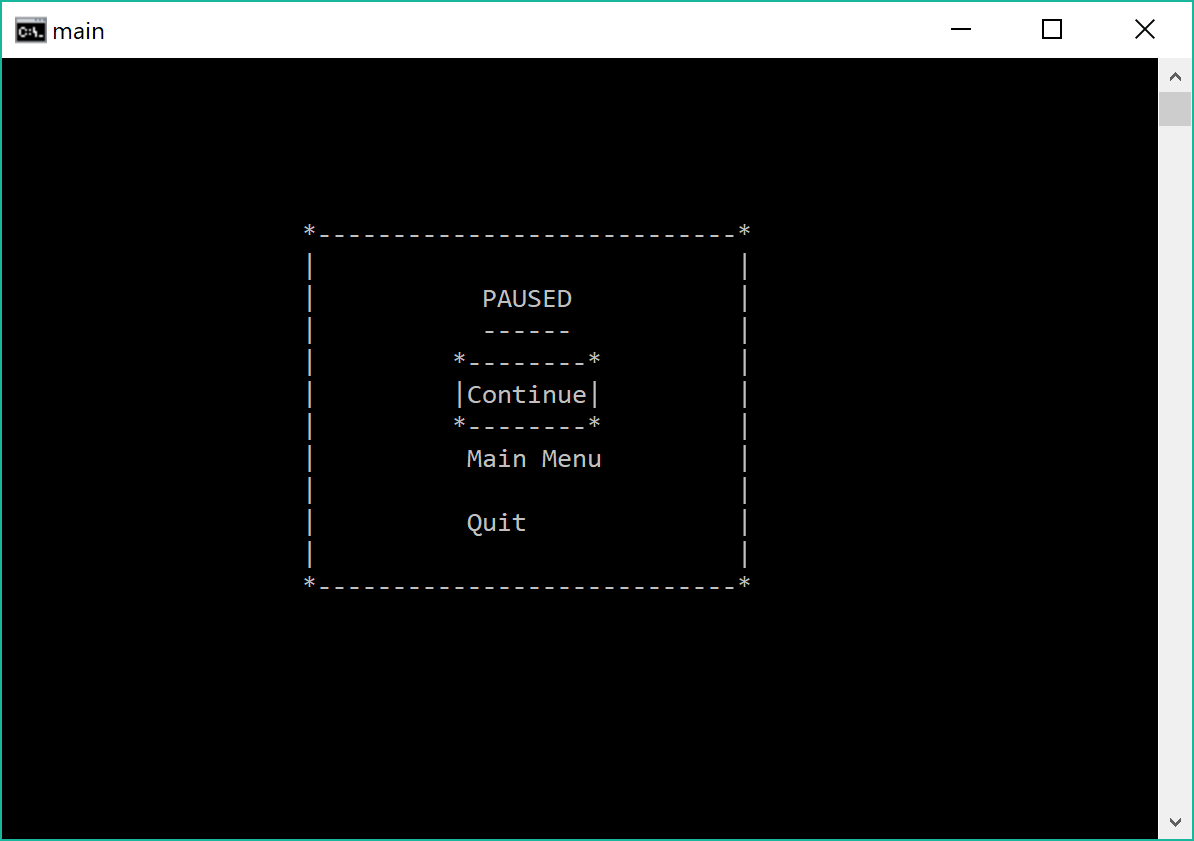
*Figure 3.3 – Screenshot of the program output – “1-Player” (player vs. computer) mode*



*Figure 3.4 – Screenshot of the program output – One player wins a round*



*Figure 3.5 – Screenshot of the program output – “2-Player” (player vs. player) mode*



*Figure 3.6 – Screenshot of the program output – Pause menu*

1. **ANALYSIS**

**Role of each member:**

* **Jim:**
* Set up the program’s layout, research on using arrow keys in displaying the program, write the menus and taking user’s inputs
* Write the 2-player mode
* Research on doing the computer’s moves
* **Kim:**
* Set up the class and its functions
* Write the 1-player mode
* Research on doing the computer’s moves

**Result Discussion:**

We were successful in creating a program that runs how it was planned and even more.

**Calculations Discussion:**

Simple addition and subtraction with respect to x and y coordinates are used to place items at specific positions on the console. Because we set the coordinates for the play field and menus ourselves, we are able to set limits telling the program how many times or how far the directional arrow keys can be pressed using “if” statements. As a directional key is pressed, either the x or y variable will increase or decrease moving the cursor. All the while variables controlling the multidimensional array are also being increased by 1 or decreased by 1 to signify where on the play field to place an X or O. This all depends on what key is pressed. This same method is used to move the cursor for choices on the menus.

1. **CONCLUSION:**

* During this project, we practiced what we have learned in class, and also learned to use array, use random function to generate a random number, the Windows libraries to use the arrow keys and to place the cursor at a specific position in the screen, system() functions to handle the screen output as well as how to use class.
* We intended to create another mode, in which the computer is “smarter” by applying Minimax Algorithm in game theory, but we did not have enough time to do research and include this part in the program.
* We used object-oriented programming in this project, but it was not the most effective, as the program is highly dependent on the arrow keys and we was not successful in finding a way to write the code more effectively.
* The program uses system() functions which does what it needs to do on Windows only, so it does not run as smoothly on other platforms.
* Besides the difficulties that has been discussed above, there are other things we can improve on, such as we should make a clearer strategy before actually starting to write the code.

1. **APPENDIX:**

(place the code)