**Report**

1. The program displays a simple tic tac toe game where the user is against a computer. The user is asked whether they would like to be ‘x’ or ‘o’ then is asked for a position to place their input ‘while the computer will then generate a random position, given it is empty, to place their input. The game will continue with each player taking turns until there is a winner or a tie game. The results of the game will then be displayed.
2. A challenge our group faced was time management. For example, everyone’s schedules are different thus, it was difficult to find a time to collaborate.

Another challenge was agreeing on a design. One group member pitched the idea to store the player and computer inputs as a string. For example, the string would be initialized to “---”, indicating empty but once the player or computer input was stored the string could change to “xo-”. I believed storing the inputs as three integer arrays to represent each row deemed much more simple, specifically when checking for a winner. I completed and tested the code following the design with my members’ code integrated. However, one of my members rejected the idea so I compromised and created a function to convert the strings into its corresponding array to ensure the initial design was still implemented. Despite my efforts and offers to fix any remaining errors, my group member still insisted on his idea and wishes for me to redo my function.

1. Throughout the project, I have become more familiarized with the syntax and structure of MIPS. I was able to gain an even deeper understanding of how computers work at a fundamental level and how computers process the code we write.

Above all, I have learned more about teamwork and collaboration. Communication is important while disagreements are inevitable. As a team, it is vital to understand compromise.

1. The game is played between player and computer, who take turns marking spaces in a three-by-three grid with their chosen symbol, X or O. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row wins the game.

The program uses the following techniques and algorithms:

* Data Section: The program initializes variables used throughout the program in the .data section. These variables include the X and O characters, the game grid, and messages to display to the user.
* Get Choice Function: The user is asked whether they would like to be ‘x’ or ‘o’.
* Print Function: The print\_array function is used to display the game grid to the user. This function takes a pointer to an array as input and prints the contents of the array in a formatted manner.
* Player Input: The askRowColumn function prompts the player to enter a row and column to mark on the grid. The function reads the input from the user and returns the values as integers.
* Insert Element: The insert\_element\_at function takes as input the row and column to insert the element, and the value to insert (1 or 2, depending on which player made the move). The function calculates the memory address of the element in the grid and inserts the value.
* Check Win: The checkWin function checks if either player has won the game. The function checks for three in a row in each of the rows, columns, and diagonals of the game grid.
* Computer Input: The computerRowColumn function generates a random row and column for the computer to mark on the grid.
* Main Loop: The main function contains the main game loop. The loop continues until one of the players wins or the game ends in a draw. Each iteration of the loop prompts the player for input, inserts the element into the grid, and checks for a win condition. If no win condition is found, the computer generates a move and the loop continues.

Overall, the program follows a simple algorithm: prompt the user for input, insert the user's move into the grid, check for a win condition, generate a computer move, insert the computer's move into the grid, and check for a win condition. The program repeats this process until a win condition is found or the game ends in a draw.

1. Contributions of Each Member

Jose

* Insert element function
* Main function
* Display board

Mon

* Getting choice of ‘x’ or ‘o’ from user
* Getting user input (functions to ask for position of row and column)
* Getting computer input

Ariel

\*\*As mentioned in the challenges section as well as the email I sent, I completed the code by integrating my members’ parts

* Check win function
* Code integration

**User Manual:**

Introduction

The Tic Tac Toe program allows you to play a game of Tic Tac Toe against the computer. The game is played on a 3x3 board where the player and computer take turns placing ‘o’ or ‘x’ on the board. The player who places three of their marks in a row, column, or diagonal wins the game.

Making Your Move

At the beginning of the game, you will be asked if you want to be ‘x’ or ‘o’. The computer will then be given the unselected choice.

To make a move, you will need to enter the row and column where you want to place your mark. The rows and columns are numbered 0, 1, and 2. A randomly generated computer input will then follow, placing their move on the board. The program will then place your mark on the board and show the updated board on the screen.

Winning the Game

The game ends when either you or the computer has placed three marks in a row, column, or diagonal, or when the board is full with no more moves available. A message will then display, indicating a winner or a draw.