PROGRAM DOCUMENTATION

Project 4

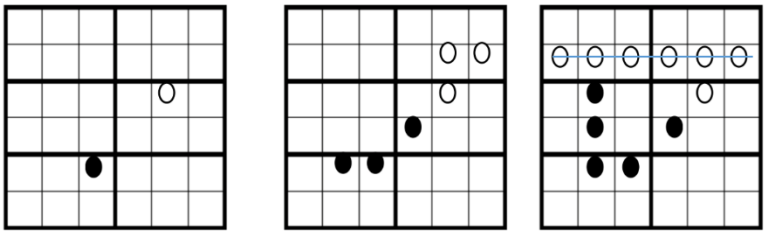
Introduction to problem solving

Course Code: CS143

Name: Mazen Tarek
ID: [redacted]

Problem Definition

A board game for 2 players, who take turns into marking spaces in a 6x6 grid. The game starts by initializing two random places for each player. Each player must place his mark near a previous typical mark. The player who succeeds in placing 6 consecutive marks of their shape in a horizontal, vertical or diagonal is the winner.



Initial Random circles | Playing | Player 1 wins

Analysis

Input: User inputs x and y coordinates (row & column) from (1 – 6 inclusive)

User is also asked to input a character either (y/n) to start the game or end.

Output: Board is drawn and marked at the specific coordinates taken from the user with his respective mark. If a player successfully gets 6 marks consecutively either horizontally, vertically or diagonally, a message popup saying which player won and asks the user to press any key, then a prompt asks if he wants to play another game.

Assumption: Player 1 is a solid circle (O) and player is (X)

Design

Libraries used:

1 – #include<stdio.h>: used for built – in functions such as scanf and printf

2 - #include<stdlib.h>: used for system (“cls”), rand(), srand() & system(“PAUSE”) and system(“COLOR F”); to display the text in a bright white color.

3 - #include<time.h>: used time in srand(time (NULL)) to generate pseudo-numbers using the processor’s internal clock to prevent similar numbers being generated in the same run from the rand() function.

Functions used:

1 – drawBoard (); Function that prints the 6x6 grid

2 - initalize_board (); Function to make all values in the array initialized = ‘ ‘;

3 - Randomize (); Function that gives both players initial random coordinates using rand () and srand () and checks if the randomized number for player 2 doesn’t produce numbers that are already marked in the grid.

4 – Player1(): Function that takes the input from the user and checks if it is in the desired range (1-6) or not it also checks if the input taken from the user is marked on the grid. And the function also checks if the input taken from the user is close to the previous input by the same user.

5 – Player2(); Same as Player1() function but it takes input from player 2 whose sign is ‘X’.

6 – checkDraw (); Function that checks if the game ended in a draw or not, if the game ended in a draw the function returns character ‘D’ if not it returns ‘ ‘

7 – checkWin (); Function that checks if a player won the game by placing 6 marks either horizontally, vertically or diagonally, if this is true the function returns the element of the array containing the character that won, else it returns ‘ ‘.

8- main (); Function used to start the code

Algorithm:

1 – A welcome screen first pops up to the user prompting him to press the enter key and (Y/y) to start the game or (N/n) to quit

2 – if the user enters (N/n) skip to last step, if the user enters (Y/y) the program enters else if condition containing a do while loop

3 – a character variable called done is initialized to ‘ ‘ , the program then initializes all elements of the 2d array to ‘ ‘ and calls the Randomize function

4 - The program then enters another do while loop, draws the board and places the mark of each player with the coordinates that got randomly produced from the Randomize function

5 – Player1 () function is then called and asks player 1 for input, if all validations rules and conditions placed in the function pass, the program then assigns the character variable done to the function checkWin ()

6 – The program then goes into an if condition which checks if done is not equal to ‘ ‘ , if the condition returns true , go to step 11 , else continue

7 – The drawBoard () is then called to display the mark placed by Player 1

8 – The programs then calls function Player2 () and asks player 2 for input, if all the conditions placed in the Player 2 () functions pass then the programs assigns the variable done to the function checkWin ()

9 - The program then goes into an if condition which checks if done is not equal to ‘ ‘ , if the condition returns true , go to step 11 , else continue

10 – function checkDraw () is called and assigned to the variable done if it returns ‘ ‘ the program continues running , else if it returns ‘D’ go to step 11

11 - the program breaks out of the do – while loop , draws the board and checks if done is equal to ‘O’ if yes a message is shown on the screen saying that player 1 won , if not an else if condition checks if done is equal to ‘X’ and a message shows that player 2 won , else a message saying that the game is a draw is printed to the screen.

12 – The system is then paused using system(“PAUSE”); until the users enters any key, when the user presses any key the screen is cleared and a message is printed asking if the user wants to play another game

13 – if the user inputs (N/n) the program breaks out of the outer do – while loop , prints a message to the user thanking him for playing then quits the program , However, if the user inputs (Y/y) the outer do – while loop continues and goes to step 3 to restart the program

Data types used:

- A 2d array of characters
- 10 integer variables, 6 of them are for input
- 2 character variables

Data types used in functions:

- 5 functions of void data type
- 2 functions with char data type

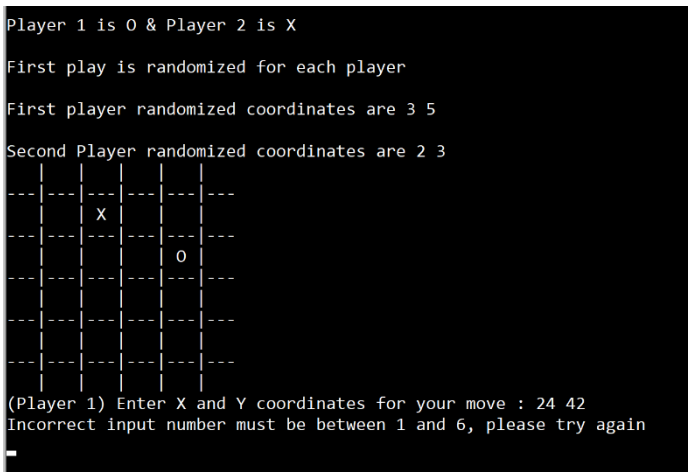
Variables used & for what reason:

- 2d array characters is a global variable, so it can be accessed throughout the whole program.
- 8 global variables including 4 global input variables for Players 1 & 2 – Player 1 (x1p,y1p), Player 2 (x2p,y2p) and the other 4 (r1,x1,r2,x2) are for storing the value generated by the Randomize function using rand for each player , so it can be accessed in Player 1 and Player 2 functions to check if input is close to the randomized number.
- 2 int variables (i,j) declared in both Player 1 and Player 2 function to be used in for loop for checking if input is close to the previous input.
- 2 char variables declared in main (done, start), done is initialized to ‘ ‘ and used as a condition in the do – while loop and start variable is used to ask the user for a character either (Y/y/N/n) to either start or end the program.

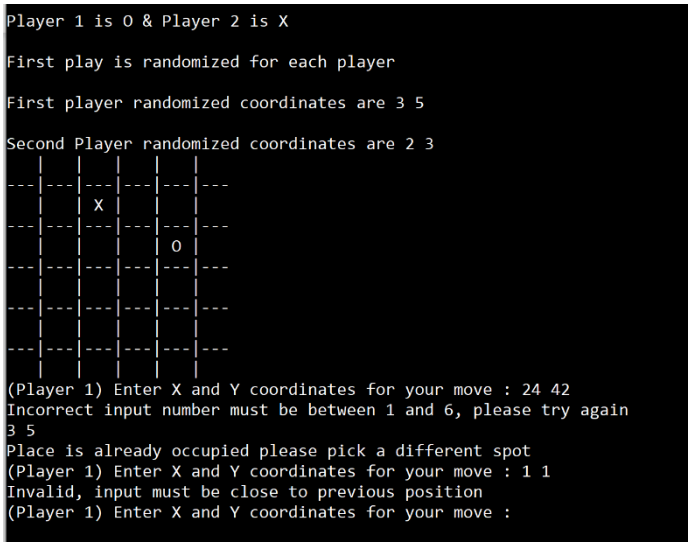
Testing

Input conditions:

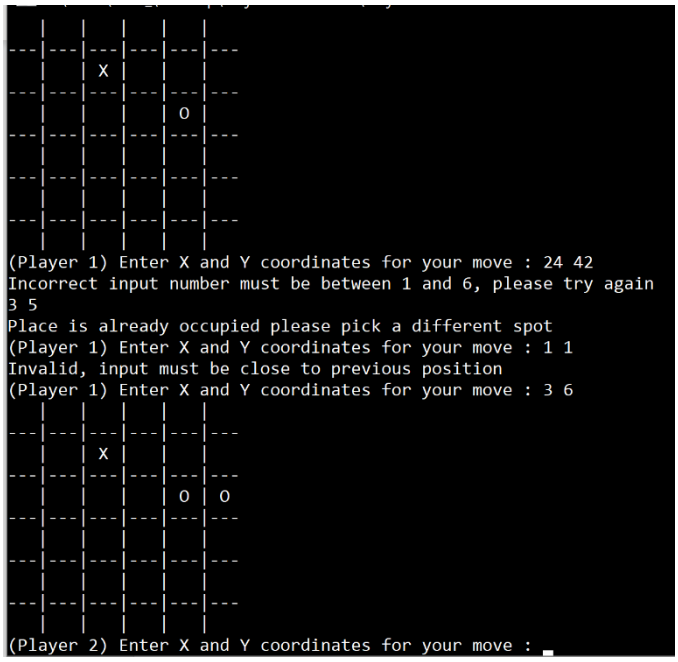
The image on the right shows the first 2 randomized coordinates for each player, and input condition is tested , when a number is entered outside of the range (1-6) inclusive, the program outputs a message saying that the input is correct and the input has to be between the specified range and it then asks the user to try again.



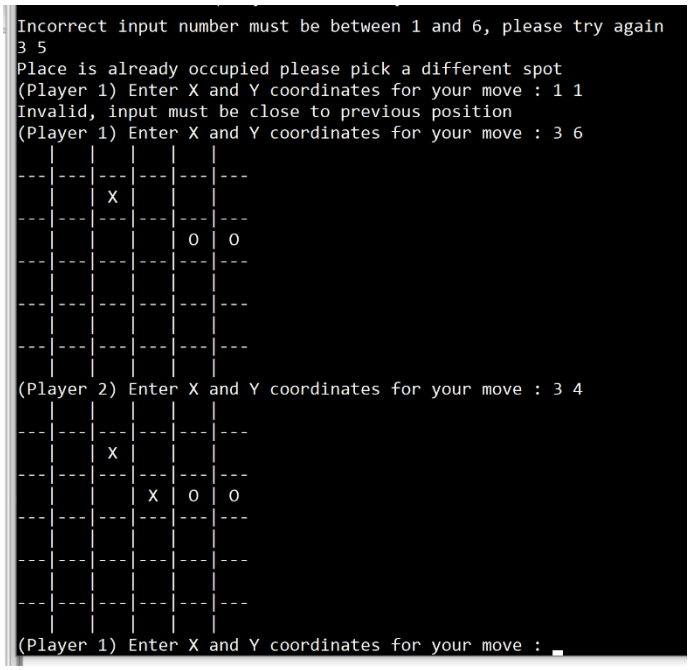
The image to the right, shows the user trying to input numbers, 3 & 5, but the program rejects the input as this place has already been marked before and asks the user for another input. The user then inputs 1 & 1 but the program also rejects it because this is far from the last input (Randomized input for player)



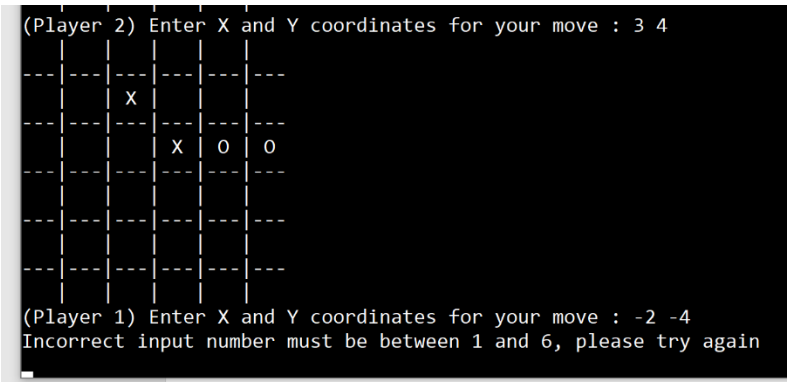
Player 1 inputs 3 & 6 and the program approves the input as it is within the range (1-6) inclusive , coordinate (3,6) was not marked and it was close to the previous position which was (3,5) , the program then asks player 2 for the input and the same conditions apply to player 2.



Player inputs coordinates 3 & 4, the program accepts the input as the input is close to the previous input diagonally, previous coordinates were (2,3)

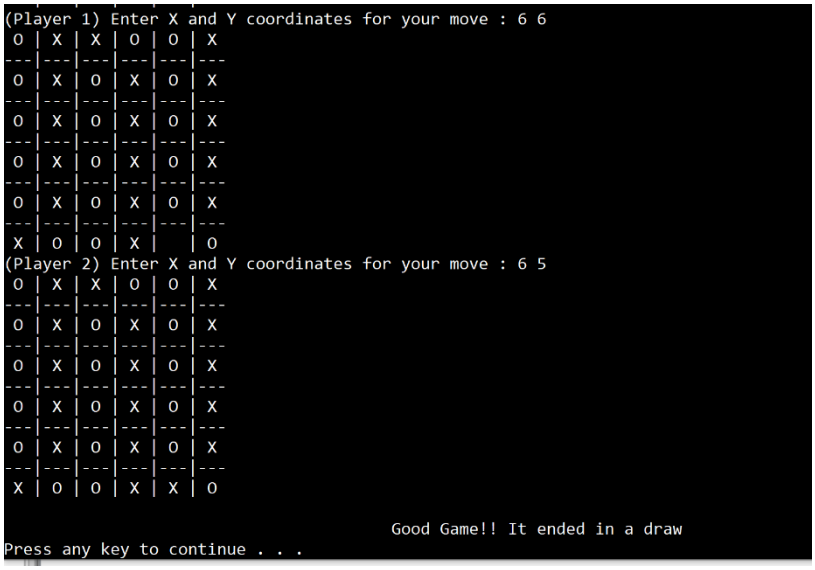


The program also restricts user input in the negative range, as it is outside the specified range (1-6) inclusive.



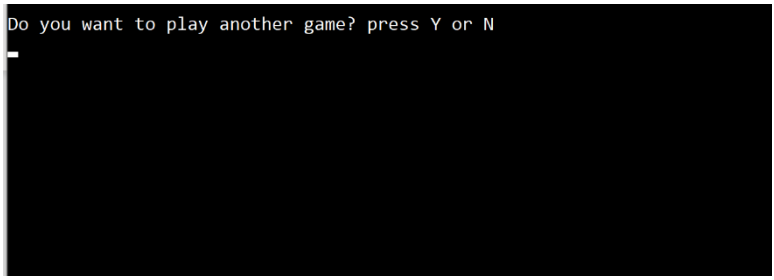
Draw Condition:

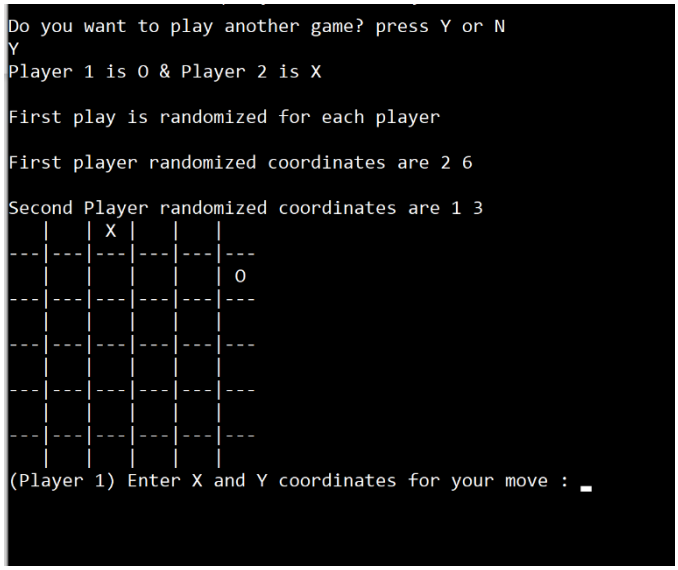
The image to the right shows the program deducing that the game ended in a draw as all the positions on the board are filled and there were no winners, so the program printed out the message (Good Game!! It ended in a draw), the system is then paused as expected using system("PAUSE") and the user is asked to press any key.



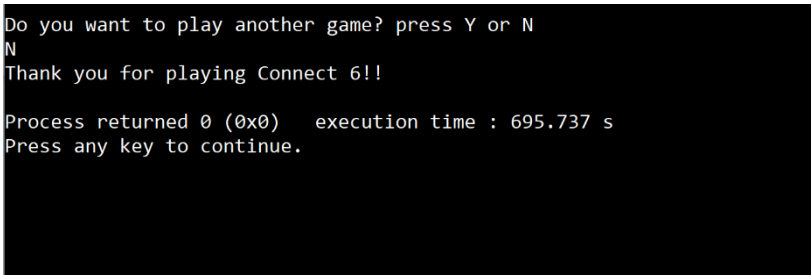
Game restart condition:

When the game ends either by a win of either players or a draw , and after the user presses any key from the system("PAUSE") function the screen is cleared using system("cls") and a message is printed asking the user if he wants to play another game, if the user inputs (Y/y) then the program restarts and produces new and different randomized values for each player using the Randomize function the following as shown in the image to the right. →





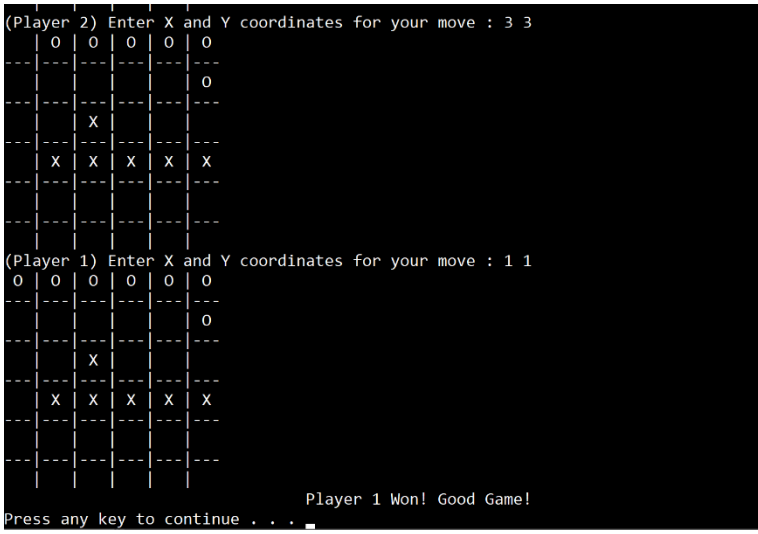
However, if the user inputs (N/n) when asked if he wants to start a new game, the program prints a message thanking the user for playing Connect 6 and does what is shown on the image to the right. →



Win Condition:

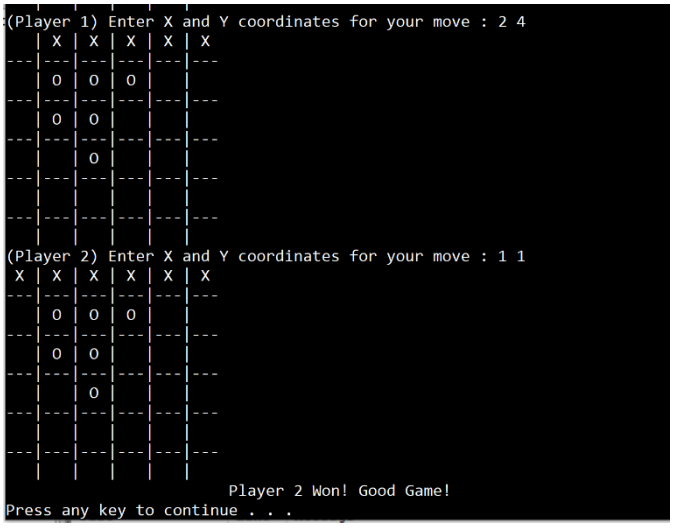
Horizontal:

As soon as any player in this case player 1 completed a full row of the same shape the program printed the message specifying which player won and paused the program and asked the user to press any key to continue, which leads to the prompt shown above asking the user if he wants to play another game



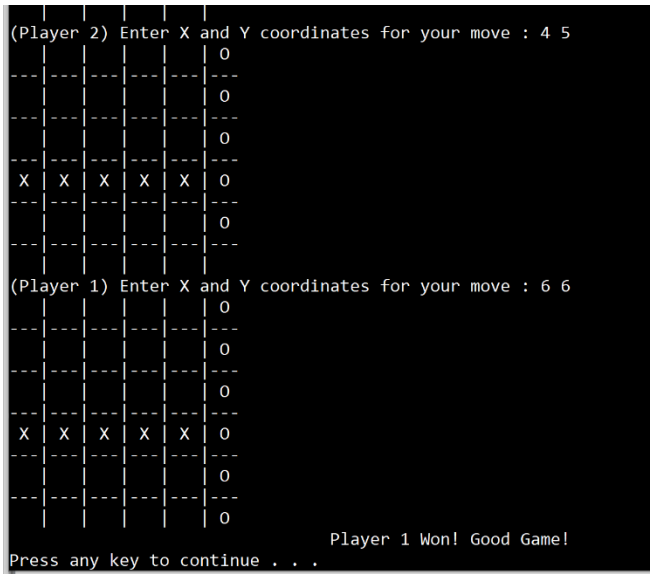
Same condition applying to Player 2

When player 2 completed a full row of the same shape 'X', the program printed a message saying player 2 won →



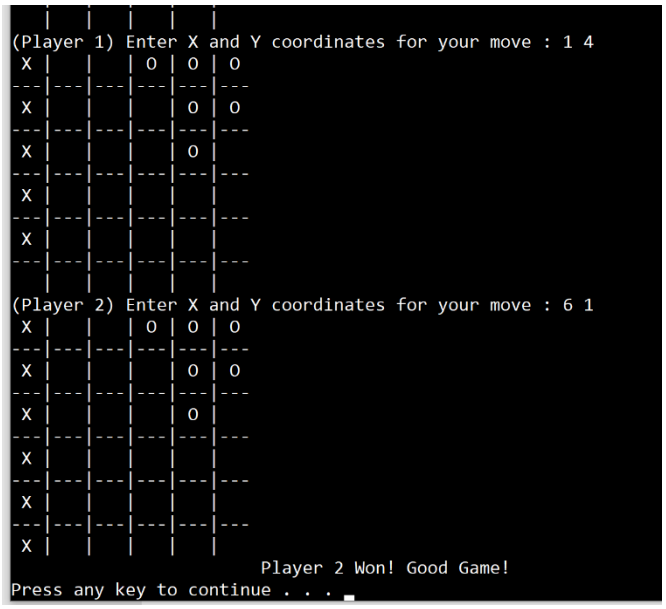
Vertical:

Same scenario to the horizontal condition as soon as one player fills 6 consecutive marks of the same shape the program says which player won and prints a message to the screen in this case the winner is player 1 ‘O’.



Same condition being applied to player 2 ‘X’ on a different column, same output showing that player 2 won after placing 6 consecutive marks

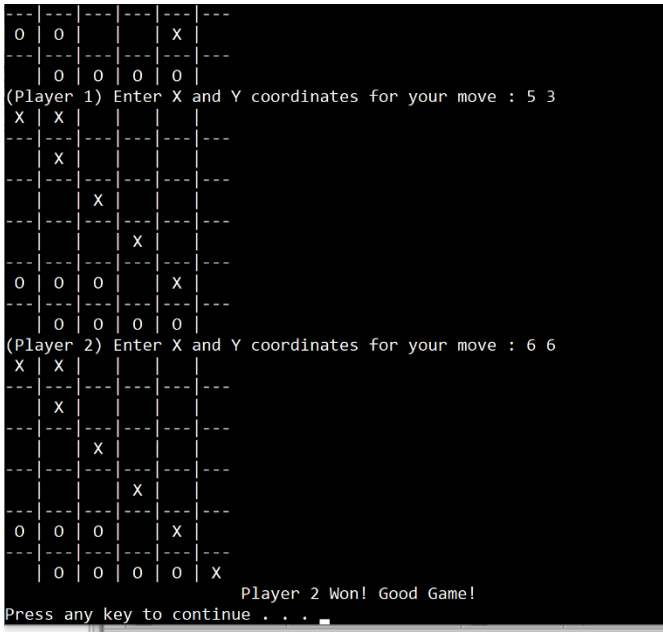
Of the same shape ‘X’.



Diagonal:

First diagonal:

As soon as a player places 6 consecutive marks in a diagonal shape the program pauses and prints a message saying who's the winner in this case player 2 'X' won as he placed 6 marks in a diagonal shape.



Second diagonal:

As soon as a player placed 6 consecutive marks of the same shape in a diagonal shape, the program displays who won and pauses , in this case player 1 won 'O' as he placed 6 marks in a diagonal shape.