Sub. Code : 21AIE111

Sub Name : Data Structure and Algorithms

Name of the Project : TIC TAC TOE USING ARRAY

Team members

1.Shyam Ganesh (21149)

2.Sidesh Sundar (21150)

3.Sabarinath (21141)

4.Jayakrishna (21154)

5.Sai Teja (21124)

6.Bharadwaj (21165)

ACKNOWLEDGEMENT

We would like to express our gratitude to
our professor R.BHUVANESWARI mam who gave us this golden opportunity to
do the project on the topic “ **TIC- TAC - TOE USING STACK“.**

We would like to extend our gratitude to the AIE department of Amrita Vishwa Vidhyapeetham, Chennai who assigned this project to us and helped us to improve our knowledge.

ABSTRACT

- Our project has been built based on “ **Tic-Tac-Toe** ”
- In the Tic-Tac-Toe game, you will see the approach of the game is implemented.
- In this game, the player have to play by typing number from 1 to 9 and X will be displayed on the box which the number represent.
- For example, if you have to select any number then for X or O will be shown on the print board, and turn for next will be there.

SOFTWARE USED

Visual Studio Code with built-in JDK.

- **Eclipse IDE.**
- **Referred sources : -**
  - ✓ GIT HUB
  - ✓ STACK OVERFLOW
  - ✓ TENSOR FLOW

CODE:

```
import java.util.Random;
import java.util.Scanner;

public class TicTacToe {

	public static void main(String[] args) {

		Scanner scanner = new Scanner(System.in);

		char[][] board = {{' ', ' ', ' '},
		                  {' ', ' ', ' '},
		                  {' ', ' ', ' '}};

		printBoard(board);

		while (true) {
			playerTurn(board, scanner);
			if (isGameFinished(board)){
				break;
			}
			printBoard(board);

			computerTurn(board);
			if (isGameFinished(board)){
				break;
			}
			printBoard(board);
		}
		scanner.close();
}


private static boolean isGameFinished(char[][] board) {

        if (hasContestantWon(board, 'X')) {
                printBoard(board);
                System.out.println("Player wins!");
                return true;
        }

        if (hasContestantWon(board, 'O')) {
                printBoard(board);
                System.out.println("Computer wins!");
                return true;
        }

        for (int i = 0; i < board.length; i++) {
                for (int j = 0; j < board[i].length; j++) {
                        if (board[i][j] == ' ') {
                                return false;
                        }
                }
        }
        printBoard(board);
        System.out.println("The game ended in a tie!");
        return true;
}
```

```
	private static boolean hasContestantWon(char[][] board, char symbol) {
		if ((board[0][0] == symbol && board [0][1] == symbol && board [0][2] ==
symbol) ||
			(board[1][0] == symbol && board [1][1] == symbol && board [1][2]
== symbol) ||
			(board[2][0] == symbol && board [2][1] == symbol && board [2][2]
== symbol) ||


			(board[0][0] == symbol && board [1][0] == symbol && board [2][0]
== symbol) ||
			(board[0][1] == symbol && board [1][1] == symbol && board [2][1]
== symbol) ||
			(board[0][2] == symbol && board [1][2] == symbol && board [2][2]
== symbol) ||


			(board[0][0] == symbol && board [1][1] == symbol && board [2][2]
== symbol) ||
			(board[0][2] == symbol && board [1][1] == symbol && board [2][0]
== symbol) ) {
			return true;
		}
		return false;
	}



	private static void computerTurn(char[][] board) {
		Random rand = new Random();
		int computerMove;
		while (true) {
			computerMove = rand.nextInt(9) + 1;
			if (isValidMove(board, Integer.toString(computerMove))) {
				break;
			}
```

```
        }
        System.out.println("Computer chose " + computerMove);
        placeMove(board, Integer.toString(computerMove), 'O');
}


private static boolean isValidMove (char[][] board, String position) {
        switch(position) {
                case "1":
                        return (board[0][0] == ' ');
                case "2":
                        return (board[0][1] == ' ');
                case "3":
                        return (board[0][2] == ' ');
                case "4":
                        return (board[1][0] == ' ');
                case "5":
                        return (board[1][1] == ' ');
                case "6":
                        return (board[1][2] == ' ');
                case "7":
                        return (board[2][0] == ' ');
                case "8":
                        return (board[2][1] == ' ');
                case "9":
                        return (board[2][2] == ' ');
                default:
                        return false;
        }
}
```

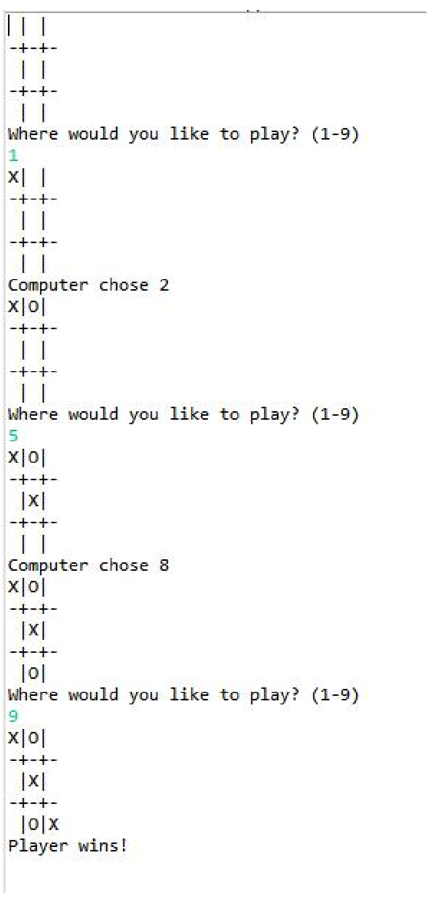
```
private static void playerTurn(char[][] board, Scanner scanner) {
        String userInput;
        while (true) {
                System.out.println("Where would you like to play? (1-9)");
                userInput = scanner.nextLine();
                if (isValidMove(board, userInput)){
                        break;
                } else {
                        System.out.println(userInput + " is not a valid move.");
                }
        }
        placeMove(board, userInput, 'X');
}


private static void placeMove(char[][] board, String position, char symbol) {
        switch(position) {
                case "1":
                        board[0][0] = symbol;
                        break;
                case "2":
                        board[0][1] = symbol;
                        break;
                case "3":
                        board[0][2] = symbol;
                        break;
                case "4":
                        board[1][0] = symbol;
                        break;
                case "5":
                        board[1][1] = symbol;
                        break;
                case "6":
                        board[1][2] = symbol;
                        break;
                case "7":
                        board[2][0] = symbol;
                        break;
                case "8":
                        board[2][1] = symbol;
                        break;
                case "9":
                        board[2][2] = symbol;
                        break;
                default:
                        System.out.println(":(");
        }
    }



        private static void printBoard(char[][] board) {
        System.out.println(board[0][0] + "|" +  board[0][1] + "|" +  board[0][2] );
        System.out.println("-+-+-");
        System.out.println(board[1][0] + "|" +  board[1][1] + "|" +  board[1][2] );
        System.out.println("-+-+-");
        System.out.println(board[2][0] + "|" +  board[2][1] + "|" +  board[2][2] );
    }
}
```

OUTPUT:



```
 | |
-+-+-
 | |
-+-+-
 | |
Where would you like to play? (1-9)
1
X| |
-+-+-
 | |
-+-+-
 | |
Computer chose 2
X|O|
-+-+-
 | |
-+-+-
 | |
Where would you like to play? (1-9)
5
X|O|
-+-+-
 |X|
-+-+-
 | |
Computer chose 8
X|O|
-+-+-
 |X|
-+-+-
 |O|
Where would you like to play? (1-9)
9
X|O|
-+-+-
 |X|
-+-+-
 |O|X
Player wins!
```

CONCLUSION:

- ❖ In the Tic-Tac-Toe game, you will see the approach of the game is implemented.
- ❖ In this game, the player have to play by typing number from 1 to 9 and X will be displayed on the box which the number represent.
- ❖ As a future scope we are planning to add GUI and few graphics can so that the game can be more attractive.