RAJAGIRI COLLEGE OF SOCIAL SCIENCES

Department Of Computer Science

Mini Project On Data Structure:

Tic Tac Toe using Linked Stack

Submitted by,

Raniya Rasheed

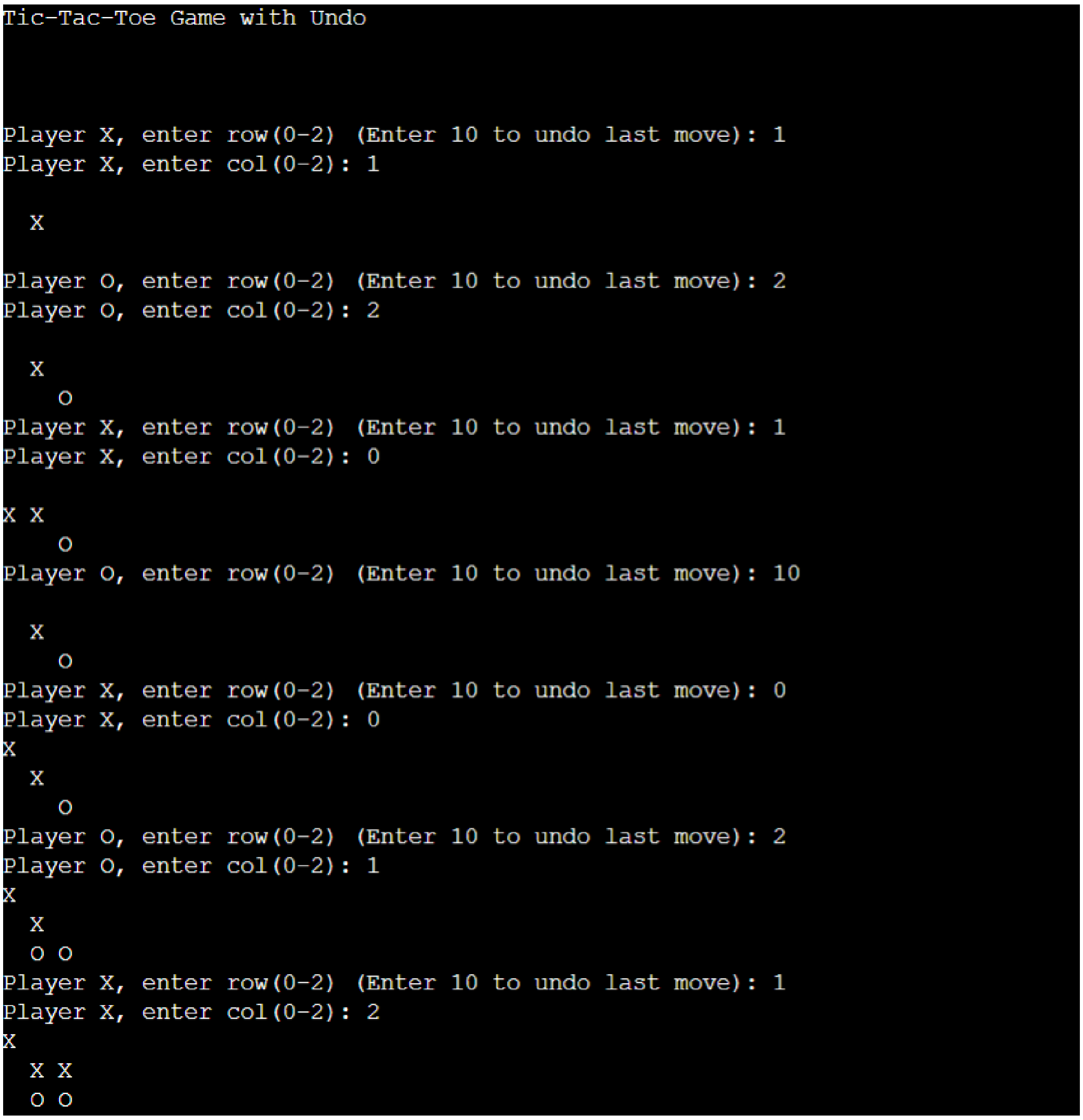
MSC CS

Roll No:25

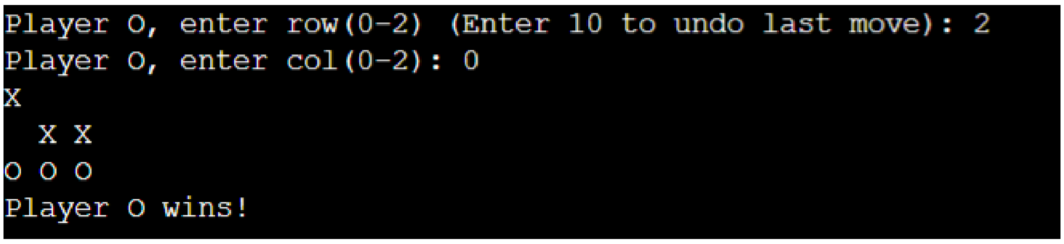
```
#include <stdio.h>
#include <stdlib.h>
#define BOARD_SIZE 3
// Structure to represent a move
struct stack{
   int row;
   int col;
   struct stack *next;
} ;
typedef struct stack stack;
stack *top=NULL;
void push(int row, int col)
{
        stack *t = (stack *)malloc(sizeof(stack));
        t->row=row;
        t->col=col;
        t->next=top;
        top=t;
}

stack * pop()
{
        stack *t=NULL;
        if(top==NULL)
                printf("Stack Underflow\n");
        else
        {
                t=top;
                top = top->next;
        }
        return t;
```

```
}
// Function to display the game board
void displayBoard(char board[BOARD_SIZE][BOARD_SIZE]) {
        int i,j;
  for ( i = 0; i < BOARD_SIZE; i++) {

    for ( j = 0; j < BOARD_SIZE; j++) {
      printf("%c ", board[i][j]);
    }
    printf("\n");
  }
}
// Function to check if a player has won
int checkWin(char board[BOARD_SIZE][BOARD_SIZE], char player) {
        int i;
  // Check rows, columns, and diagonals for a win
  for ( i = 0; i < BOARD_SIZE; i++) {
    if ((board[i][0] == player && board[i][1] == player && board[i][2] == player) ||
      (board[0][i] == player && board[1][i] == player && board[2][i] == player)) {
      return 1; // Player wins
    }
  }
  if ((board[0][0] == player && board[1][1] == player && board[2][2] == player) ||
    (board[0][2] == player && board[1][1] == player && board[2][0] == player)) {
    return 1; // Player wins
  }
  return 0; // No winner yet
}
int main() {
  char board[BOARD_SIZE][BOARD_SIZE] = {
    {' ', ' ', ' '},
        {' ', ' ', ' '},
        {' ', ' ', ' '}
    };
    char currentPlayer = 'X';
    int row, col;
    int turn;
            stack *temp;
    printf("Tic-Tac-Toe Game with Undo\n");
    for ( turn = 0; turn < BOARD_SIZE * BOARD_SIZE; turn++) {
        displayBoard(board);
        printf("Player %c, enter row(0-%d) (Enter 10 to undo last move): ", currentPlayer, BOARD_SIZE -
1);
        scanf("%d", &row);
        if(row == 10)
        {
            if (top != NULL) {
                temp = pop();
                board[temp->row][temp->col] = ' ';
                currentPlayer = (currentPlayer == 'X') ? 'O' : 'X';
                continue;
            } else {
                printf("No moves to undo.\n");
                turn--; // Repeat this turn
                continue;
            }
        }
        else
        {
            printf("Player %c, enter col(0-%d): ", currentPlayer, BOARD_SIZE - 1);
            scanf("%d",&col);
            // Check if the chosen cell is valid
        if (row < 0 || row >= BOARD_SIZE || col < 0 || col >= BOARD_SIZE || board[row][col] != ' ')
        {
            printf("Invalid move. Try again.\n");
            turn--; // Repeat this turn
            continue;
        }
        // Push the current move onto the stack
        push(row, col);
        board[row][col] = currentPlayer;
        if (checkWin(board, currentPlayer)) {
            displayBoard(board);
            printf("Player %c wins!\n", currentPlayer);
            break;
        }

        // Switch to the other player
            currentPlayer = (currentPlayer == 'X') ? 'O' : 'X';
        }
    }
    if (!checkWin(board, 'X') && !checkWin(board, 'O')) {
        displayBoard(board);
        printf("It's a draw!\n");
    }
    return 0;
}
```

OUTPUT



```
Tic-Tac-Toe Game with Undo



Player X, enter row(0-2) (Enter 10 to undo last move): 1
Player X, enter col(0-2): 1

  X

Player O, enter row(0-2) (Enter 10 to undo last move): 2
Player O, enter col(0-2): 2

  X
    O
Player X, enter row(0-2) (Enter 10 to undo last move): 1
Player X, enter col(0-2): 0

X X
    O
Player O, enter row(0-2) (Enter 10 to undo last move): 10

  X
    O
Player X, enter row(0-2) (Enter 10 to undo last move): 0
Player X, enter col(0-2): 0
X
  X
    O
Player O, enter row(0-2) (Enter 10 to undo last move): 2
Player O, enter col(0-2): 1
X
  X
  O O
Player X, enter row(0-2) (Enter 10 to undo last move): 1
Player X, enter col(0-2): 2
X
  X X
  O O
```



```
Player O, enter row(0-2) (Enter 10 to undo last move): 2
Player O, enter col(0-2): 0
X
  X X
O O O
Player O wins!
```