مينا نسيم إدوار نسيم يوسف

سيكشن 7

ثالثه حاسبات ونظم تحكم

```
#include <stdio.h>
#define EMPTY ' '
#define PLAYER_X 'X'
#define PLAYER_O 'O'
char grid[3][3] = {
{EMPTY, EMPTY, EMPTY},
{EMPTY, EMPTY, EMPTY},
{EMPTY, EMPTY, EMPTY},
};
int isGameOver(char grid[][3]) {
// Check rows
for (int i = 0; i < 3; i++) {
if (grid[i][0] == grid[i][1] && grid[i][1] == grid[i][2] && grid[i][0] != EMPTY) {
return 1; // Winner found
}
}
// Check columns
for (int j = 0; j < 3; j++) {
if (grid[0][j] == grid[1][j] && grid[1][j] == grid[2][j] && grid[0][j] != EMPTY) {
return 1; // Winner found
}
}
// Check diagonals
if (grid[0][0] == grid[1][1] && grid[1][1] == grid[2][2] && grid[0][0] != EMPTY) {
return 1; // Winner found
```

```
 }
 if (grid[0][2] == grid[1][1] && grid[1][1] == grid[2][0] && grid[0][2] != EMPTY) {
 return 1; // Winner found
 }
 // Check for draw (all cells filled)
 for (int i = 0; i < 3; i++) {
 for (int j = 0; j < 3; j++) {
 if (grid[i][j] == EMPTY) {
 return 0; // Game not over, empty cells present
 }
 }
 }
 return 2; // Draw
}

void printGrid(char grid[][3]) {
 for (int i = 0; i < 3; i++) {
 for (int j = 0; j < 3; j++) {
 printf("%c ", grid[i][j]);
 }
 printf("\n");
 }
}
int main() {
 char currentPlayer = PLAYER_X;
 int gameOver = 0;
 while (!gameOver) {
 printGrid(grid);
 printf("Turn: %c\n", currentPlayer);
```

```
int index;
// Get valid input from the user
do {
printf("Enter an index (1-9): ");
scanf("%d", &index);
} while (index < 1 || index > 9 || grid[(index - 1) / 3][(index - 1) % 3] != EMPTY);
// Decrement index to match array indexing (0-based)
index--;
// Update grid
grid[index / 3][index % 3] = currentPlayer;
gameOver = isGameOver(grid);
if (gameOver == 1) {
printGrid(grid);
printf("Player %c wins!\n", currentPlayer);
} else if (gameOver == 2) {
printf("It's a draw!\n");
printf("Starting new game...\n");
// Reset grid
for (int i = 0; i < 3; i++) {
for (int j = 0; j < 3; j++) {
grid[i][j] = EMPTY;
}
}
}
// Switch player for next turn
currentPlayer = (currentPlayer == PLAYER_X) ? PLAYER_O : PLAYER_X;
}
return 0;
}
```

```c
#include <stdio.h>
#define EMPTY ' '
#define PLAYER_X 'X'
#define PLAYER_O 'O'
char grid[3][3] = {
 {EMPTY, EMPTY, EMPTY},
 {EMPTY, EMPTY, EMPTY},
 {EMPTY, EMPTY, EMPTY},
};
int isGameOver(char grid[][3]) {
 // Check rows
 for (int i = 0; i < 3; i++) {
 if (grid[i][0] == grid[i][1] && grid[i][1] == grid[i][2] && grid[i][0] != EMPTY) {
 return 1; // Winner found
 }
 }
 // Check columns
 for (int j = 0; j < 3; j++) {
 if (grid[0][j] == grid[1][j] && grid[1][j] == grid[2][j] && grid[0][j] != EMPTY) {
 return 1; // Winner found
 }
 }
 // Check diagonals
 if (grid[0][0] == grid[1][1] && grid[1][1] == grid[2][2] && grid[0][0] != EMPTY) {
 return 1; // Winner found
 }
 if (grid[0][2] == grid[1][1] && grid[1][1] == grid[2][0] && grid[0][2] != EMPTY) {
 return 1; // Winner found
 }
 // Check for draw (all cells filled)
 for (int i = 0; i < 3; i++) {
 for (int j = 0; j < 3; j++) {
 if (grid[i][j] == EMPTY) {
 return 0; // Game not over, empty cells present
 }
 }
 }
 return 2; // Draw
}

void printGrid(char grid[][3]) {
 for (int i = 0; i < 3; i++) {
 for (int j = 0; j < 3; j++) {
 printf("%c ", grid[i][j]);
 }
 printf("\n");
 }
}
int main() {
 char currentPlayer = PLAYER_X;
 int gameOver = 0;
 while (!gameOver) {
 printGrid(grid);
 printf("Turn: %c\n", currentPlayer);
 int index;
 // Get valid input from the user
 do {
 printf("Enter an index (1-9): ");
 scanf("%d", &index);
 } while (index < 1 || index > 9 || grid[(index - 1) / 3][(index - 1) % 3] != EMPTY);
 // Decrement index to match array indexing (0-based)
 index--;
 // Update grid
 grid[index / 3][index % 3] = currentPlayer;
 gameOver = isGameOver(grid);
 if (gameOver == 1) {
 printGrid(grid);
 printf("Player %c wins!\n", currentPlayer);
 } else if (gameOver == 2) {
 printf("It's a draw!\n");
 printf("Starting new game...\n");
 // Reset grid
 for (int i = 0; i < 3; i++) {
 for (int j = 0; j < 3; j++) {
 grid[i][j] = EMPTY;
 }
 }
 }
 // Switch player for next turn
 currentPlayer = (currentPlayer == PLAYER_X) ? PLAYER_O : PLAYER_X;
 }
 return 0;
}
```

Text logs from terminal when the program is working in the following condition

■ (X) win the game

Turn: X

Enter an index (1-9): 1

X

Turn: O

Enter an index (1-9): 2

X O

Turn: X

Enter an index (1-9): 4

X O

X

Turn: O

Enter an index (1-9): 3

X O O

X

Turn: X

Enter an index (1-9): 7

X O O

X

X

Player X wins!

■ (O) win the game

Turn: X

Enter an index (1-9): 2

X

Turn: O

Enter an index (1-9): 3

X O

Turn: X

Enter an index (1-9): 1

X X O

Turn: O

Enter an index (1-9): 5

X X O

O

Turn: X

Enter an index (1-9): 6

X X O

O X

Turn: O

Enter an index (1-9): 7

X X O

O X

O

Player O wins!

■ (X) win the game after 2 draw condition.

Turn: X

Enter an index (1-9): 1

X

Turn: O

Enter an index (1-9): 4

X

O

Turn: X

Enter an index (1-9): 3

X X

O

Turn: O

Enter an index (1-9): 7

X X

O

O

Turn: X

Enter an index (1-9): 5

X X

O X

O

Turn: O

Enter an index (1-9): 6

X X

O X O

O

Turn: X

Enter an index (1-9): 8

X X

O X O

O X

Turn: O

Enter an index (1-9): 9

X X

O X O

O X O

Turn: X

Enter an index (1-9): 2

X X X

O X O

O X O

Player X wins!

■ (O) wins the game after 1 draw condition.

Turn: X

Enter an index (1-9): 1

X

Turn: O

Enter an index (1-9): 2

X O

Turn: X

Enter an index (1-9): 3

X O X

Turn: O

Enter an index (1-9): 5

X O X

O

Turn: X

Enter an index (1-9): 6

X O X

O X

Turn: O

Enter an index (1-9): 8

X O X

O X

O

Player O wins!