Программа, представленная в файлах Program.cs и Game.cs, реализует игру "Крестики-нолики" для двух игроков в консоли.

Файл Program.cs содержит метод Main, который является точкой входа в программу. В методе Main создается экземпляр класса Game и вызывается метод Start, чтобы начать игру. Таким образом, основная задача этого файла - запустить игру и управлять ее ходом.

Файл Game.cs содержит класс Game, который представляет игровую логику и методы для управления игрой. Методы я указала цветом и похожим цветом описание этого метода(рядом).

В файле Program.cs:

using System;

namespace TicTacToe

{

class Program

{

static void Main(string[] args)

{

Game game = new Game();

game.Start();

}

}

}

В файле Game.cs:

using System;

namespace TicTacToe

{

class Game

{

private char[,] board;

private char currentPlayer;

private bool isGameFinished;

**Start(): Этот метод инициализирует игровое поле, устанавливает начального**

**игрока, и запускает основной игровой цикл. В цикле выводится текущее**

**состояние игрового поля, игрок делает ход, проверяется результат игры, и**

**игроки переключаются до тех пор, пока игра не завершится.**

public void Start()

{

InitializeBoard();

currentPlayer = 'X';

isGameFinished = false;

while (!isGameFinished)

{

PrintBoard();

MakeMove();

CheckGameResult();

SwitchPlayers();

}

}

InitializeBoard(): Этот метод инициализирует игровое поле (матрицу

3x3) пустыми значениями. Каждый элемент матрицы устанавливается в

символ пробела ' '.

private void InitializeBoard()

{

board = new char[3, 3];

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

board[i, j] = ' ';

}

}

}

private void PrintBoard()

{

PrintBoard(): Этот метод выводит текущее

состояние игрового поля в консоль. Он очищает

консоль, затем выводит номера столбцов и

значения каждой ячейки игрового поля.

Console.Clear();

Console.WriteLine(" 0 1 2");

for (int i = 0; i < 3; i++)

{

Console.Write(i + " ");

for (int j = 0; j < 3; j++)

{

Console.Write(board[i, j] + " ");

}

Console.WriteLine();

}

}

Этот метод позволяет текущему игроку сделать ход. Он запрашивает ввод игрока для указания координат хода (строка и столбец), разделяя ввод пробелом. Введенные значения преобразуются в целые числа и проверяются на корректность. Если ход некорректный (выходит за границы поля или ячейка уже занята), выводится сообщение об ошибке и метод MakeMove() вызывается рекурсивно, чтобы игрок ввел корректные значения. Если ход корректный, символ текущего игрока добавляется в соответствующую ячейку игрового поля.

private void MakeMove()

{

Console.WriteLine($"Игрок {currentPlayer}, сделайте свой ход (столбец строки:");

string[] input = Console.ReadLine().Split(' ');

int row = int.Parse(input[0]);

int col = int.Parse(input[1]);

if (row < 0 || row >= 3 || col < 0 || col >= 3 || board[row, col] != ' ')

{

Console.WriteLine("Неверный ход! Попробуйте еще раз.");

MakeMove();

}

else

{

board[row, col] = currentPlayer;

}

}

Этот метод проверяет текущее состояние игры на наличие победителя или ничью. Он проверяет все возможные комбинации победы: строки, столбцы и диагонали. Если одна из комбинаций совпадает с символом текущего игрока, игра считается завершенной и выводится сообщение о победе этого игрока. Если все ячейки игрового поля заполнены и нет победителя, игра считается ничьей.

private void CheckGameResult():

{

for (int i = 0; i < 3; i++)

{

if (board[i, 0] == currentPlayer && board[i, 1] == currentPlayer && board[i, 2] == currentPlayer)

{

isGameFinished = true;

Console.WriteLine($"Игрок {currentPlayer} побеждает!");

return;

}

}

// Check columns

for (int j = 0; j < 3; j++)

{

if (board[0, j] == currentPlayer && board[1, j] == currentPlayer && board[2, j] == currentPlayer)

{

isGameFinished = true;

Console.WriteLine($"Игрок {currentPlayer} побеждает!");

return;

}

}

// Check diagonals

if ((board[0, 0] == currentPlayer && board[1, 1] == currentPlayer && board[2, 2] == currentPlayer) ||

(board[0, 2] == currentPlayer && board[1, 1] == currentPlayer && board[2, 0] == currentPlayer))

{

isGameFinished = true;

Console.WriteLine($"Игрок {currentPlayer} побеждает!");

return;

}

// Check for a draw

bool isBoardFull = true;

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

if (board[i, j] == ' ')

{

isBoardFull = false;

break;

}

}

}

if (isBoardFull)

{

isGameFinished = true;

Console.WriteLine("Ничья");

}

SwitchPlayers(): Этот метод переключает текущего игрока. Если текущий игрок 'X', он

меняется на 'O', и наоборот.

}

private void SwitchPlayers()

{

currentPlayer = currentPlayer == 'X' ? 'O' : 'X';

}

}

}