**\* Крестики-нолики**

Вам с Васей наконец-то надоело тренироваться на маленьких программках и вы взялись за настоящее дело! Вы решили написать игру [крестики-нолики](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B5%D1%81%D1%82%D0%B8%D0%BA%D0%B8-%D0%BD%D0%BE%D0%BB%D0%B8%D0%BA%D0%B8)!

Начать было решено с подпрограммы, определяющей не закончилась ли уже игра, а если закончилась, то кто выиграл.

Методу GetGameResult передается поле, представленное массивом 3х3 из enum Markers. Вам надо вернуть победителя CrossWin или CircleWin, если таковой имеется или Draw, если выигрышной последовательности нет ни у одного, либо есть у обоих.

Постарайтесь придумать красивое, понятное решение.

Подумайте, как разбить задачу на более простые подзадачи. Попытайтесь выделить один или два вспомогательных метода.

Если вы в затруднении, воспользуйтесь подсказками (кнопка Get hint)

public enum Mark

{

Empty,

Cross,

Circle

}

public enum GameResult

{

CrossWin,

CircleWin,

Draw

}

public static void Main()

{

Run("XXX OO. ...");

Run("OXO XO. .XO");

Run("OXO XOX OX.");

Run("XOX OXO OXO");

Run("... ... ...");

Run("XXX OOO ...");

Run("XOO XOO XX.");

Run(".O. XO. XOX");

}

private static void Run(string description)

{

Console.WriteLine(description.Replace(" ", Environment.NewLine));

Console.WriteLine(GetGameResult(CreateFromString(description)));

Console.WriteLine();

}

private static Mark[,] CreateFromString(string str)

{

var field = str.Split(' ');

var ans = new Mark[3, 3];

for (int x = 0; x < field.Length; x++)

for (var y = 0; y < field.Length; y++)

ans[x, y] = field[x][y] == 'X' ? Mark.Cross : (field[x][y] == 'O' ? Mark.Circle : Mark.Empty);

return ans;

}

public static GameResult GetGameResult(Mark[,] field)

{

}

**Программа:**

using System;

namespace umop5o16TicTacToe

{

class Program

{

public enum Mark

{

Empty,

Cross,

Circle

}

public enum GameResult

{

CrossWin,

CircleWin,

Draw

}

public static void Main()

{

Run("XXX OO. ...");

Run("OXO XO. .XO");

Run("OXO XOX OX.");

Run("XOX OXO OXO");

Run("... ... ...");

Run("XXX OOO ...");

Run("XOO XOO XX.");

Run(".O. XO. XOX");

Console.ReadKey();

}

private static void Run(string description)

{

Console.WriteLine(description.Replace(" ", Environment.NewLine));

Console.WriteLine(GetGameResult(CreateFromString(description)));

Console.WriteLine();

}

private static Mark[,] CreateFromString(string str)

{

var field = str.Split(' ');

var ans = new Mark[3, 3];

for (int x = 0; x < field.Length; x++)

for (var y = 0; y < field.Length; y++)

ans[x, y] = field[x][y] == 'X' ? Mark.Cross : (field[x][y] == 'O' ? Mark.Circle : Mark.Empty);

return ans;

}

public static GameResult GetGameResult(Mark[,] field)

{

if (HasWinSequence(field, Mark.Cross) & HasWinSequence(field, Mark.Circle))

return GameResult.Draw;

if (HasWinSequence(field, Mark.Cross))

return GameResult.CrossWin;

if (HasWinSequence(field, Mark.Circle))

return GameResult.CircleWin;

return GameResult.Draw;

}

public static bool HasWinSequence(Mark[,] field, Mark mark)

{

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

{

if (field[i, 0] == mark && field[i, 1] == mark && field[i, 2] == mark)

return true;

if (field[0, i] == mark && field[1, i] == mark && field[2, i] == mark)

return true;

}

if (field[0, 0] == mark && field[1, 1] == mark && field[2, 2] == mark)

return true;

return field[0, 2] == mark && field[1, 1] == mark && field[2, 0] == mark;

}

}

}