**\* Tic-tac-toe**

You and Vasya are finally tired of training on small programs and you got down to real business! You've decided to write a tic-tac-toe game!

It was decided to start with a subroutine that determines whether the game has already ended, and if it ended, then who won.

The GetGameResult method is passed a field represented by a 3x3 array from enum Markers. You need to return the winner of CrossWin or CircleWin, if there is one, or Draw, if neither has a winning sequence, or both.

Try to come up with a nice, understandable solution.

Consider how to break a task into simpler sub-tasks. Try to highlight one or two helper methods.

If you are in difficulty, use the hints (Get hint button)

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)

{

}

**Code:**

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;

}

}

}