**Практика «Пустой лабиринт»**

[Скачайте проект Mazes](https://ulearn.me/Exercise/StudentZip?courseId=BasicProgramming&slideId=6dd79ed2-2d6e-487f-88dc-7414a15600fa) и изучите его. Там заготовлены несколько фиксированных лабиринтов. В каждом лабиринте вам нужно довести робота до выхода — клетки, помеченной зеленым кружком.

Сам лабиринт не известен и у робота нет сенсоров, чтобы его исследовать, но известен тип лабиринта и его размеры.

Этого достаточно, чтобы построить маршрут и отдать нужные команды роботу.

В этой задаче лабиринт — это пустая комната, окружённая стеной по периметру, в которой из верхнего левого угла с координатами (1, 1) нужно пройти в правый нижний с координатами (width-2, height-2).

Дополнительные ограничения:

1. Запрещено использовать более одного цикла в одном методе.
2. Запрещено иметь методы длиннее 6 строк кода.
3. Запрещено использовать ключевое слово catch
4. Разрешено создавать вспомогательные методы, но только понятными именами, в том числе именами аргументов.

**О понятности методов**

Важно уметь создавать такие методы, чтобы они были понятны без необходимости заглядывать внутрь метода. При работе над большим проектом это позволит вам и вашим коллегам быстрее ориентироваться в коде.

Например, такой метод непонятен, потому что неясно, ни что такое i, ни куда будет двигаться робот:

void Move(Robot robot, int i)

А такой метод плох тем, что вводит в заблуждение читателя:

void MoveLeft(Robot robot, int stepCount)

{

for (int i = 0; i < stepCount - 2; i++)

robot.MoveTo(Direction.Left);

}

Ведь в сигнатуре метода он обещает, что сделает stepCount шагов, а вместо этого делает stepCount-2 шага.

Потренируйтесь создавать понятные методы на данной серии задач.

// Вставьте сюда финальное содержимое файла EmptyMazeTask.cs

namespace Mazes

{

public static class EmptyMazeTask

{

public static void MoveOut(Robot robot, int width, int height)

{

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

{ robot.MoveTo(Direction.Right); }

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

{ robot.MoveTo(Direction.Down); }

}

}

}