Once upon a time, in a galaxy far far away, there lived a monster. His name was Mitko

Mitko lived in a labyrinth and usually could go through the walls. Unfortunately one morning it was so hungry that it had to turn off this ability in order to save energy. Now he needed to find food in the labyrinth and hopefully to use the shortest possible way. Oh, and one more thing – the labyrinth is shifting and each morning its different.

Here is how your program will read an example labyrinth entry:

	A	B	C	D	E	F	G	H	I	J	K	L	M
1													
2								N	N	N			
3			N	N			N	N		N			
4				N						N	N	N	
5				N				X		N			
6	M			N	N	N	N	N	N	N			
7													
8													

What does it mean?
Empty space – free pass corridor
N – hard wall
M – This is Mitko, who needs your help
X – This is the food that you need to get

Possible moves:

Mitko can move in each direction – north (up), south(down), east(right), west(left)

Make a program that reads a matrix of symbols, that represent a labyrinth and finds the optimal path from a given starting position to a given target position.

The program will get a path to a file (csv format) that contains the labyrinth. You will get the position of Mitko (x and y) and the position of the food (x and y)

At the end you should have a list of steps, that Mitko should go through to get to the food.

Here is example Main function:

```
public static void main (String[] args) {
        Maze maze = new Maze(args[0]);
        Point start = new Point(args[1], args[2]));
        Point end = new Point(args[3], args[4]));

        List<Point> path = maze.findPath(start, end);

        for (Point step : path) {
                System.out.print(step.toString());

                if (!step.equals(end)) {
                        System.out.print(" ,");
                }
        }

}
```

Be careful – make sure you find an optimal path. This is a solution, but not optimal:

	A	B	C	D	E	F	G	H	I	J	K	L	M
1							*	*	*	*	*		
2						*	*	N	N	N	*		
3			N	N		*	N	N		N	*	*	*
4				N		*	*	*		N	N	N	*
5				N				X		N	*	*	*
6	M			N	N	N	N	N	N	N	*		
7	*	*	*	N	*	*	*	*	*	*	*		
8			*	*	*								

[(A, 7), (B, 7), (C, 7), (C, 8) (D, 8), (E, 8), (F, 7), (G, 7), (H, 7), (I, 7), (J, 7), (K, 7), (K, 6), (K, 5), (L, 5), (M, 5), (M, 4), (M, 3), (L, 3), (K, 3), (K, 2), (K, 1), (J, 1), (I, 1), (H, 1), (G,1), (G, 2), (F, 2), (F, 3), (F, 4), (G, 4), (H, 4)]