Take the Right Turn!

written by Lisa Wang

Please read the student handout first. Thanks!

Concepts Covered:

- Recursive Backtracking (Variation of the maze problem)

Why is this problem relevant?

With rising gasoline prices and the cost of time, delivery companies like UPS or FedEx are employing sophisticated algorithms to make the routes for their drivers more efficient by reducing left turns. On average, left turns generally take longer than right turns, since drivers have to wait for oncoming traffic to clear and for the traffic light to be green in order to turn left. Right turns, on the other hand, can be made even on red at most intersections.

By shortening the time the trucks have to wait for a left turn, the new algorithms save gasoline and reduce carbon emissions. In addition, packages are delivered faster and reducing left turns can even make traffic safer. For UPS, the simple idea of preferring right turns over left turns has saved millions of dollars.

Today's Challenge:
with solutions for parts a), b) and c)

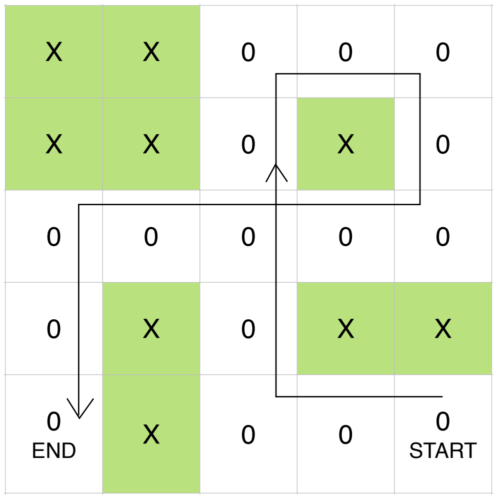
You are given a world, represented as a char grid where '0' stands for a cell that is part of a street, and 'X' stands for a cell that is part of a building.

X	X	0	0	0
X	X	0	X	0
0	0	0	0	0
0	X	0	X	X
0 END	X	0	0	0 START

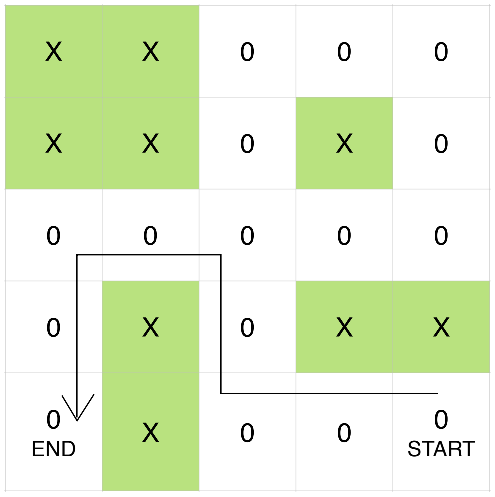
a) Find the cheapest path from start to end.
b) Find the shortest path from start to end.

	Cost	Length
Cheapest Path	**34**	**16**
Shortest Path	**37**	**8**

Cheapest Path:



Shortest Path:



c) Similarities and Differences between the section problem and the maze problem from the 106B/X Course Reader (p. 397):

Similarities:
For both problems, we can use recursive backtracking. We have to mark the cells we have visited so far on the path. When we come back to examine a different path, we have to unmark those cells.

Differences:

Maze Problem	**Cheapest Path Problem**
Every cell along a path is visited exactly once.	Since a path can intersect itself, every cell on a path is either visited once or twice. We can create a grid of markers, that stores for each cell how many times it has been visited so far.
We assume that there are no loops in the maze.	There could be loops. In fact, without loops, one could never take right turns instead of a left turn, so the problem would not make sense.
We only care about finding some path that brings us outside the maze. Therefore, we can return as soon as we find such a path.	We have to find and compare every possible path from start to end, since we want to find the cheapest path. Therefore, we have to keep track of the cheapest path and update it if we have found a cheaper one.

To implement the function `findCheapestPath`, the students should refer to the data structures, enum types and helper functions on the last two pages of the student handout.