Binary Maze

Given an R x C matrix where each element can either be 0 or 1, find the shortest path between a given source node to a destination node. The path can only be created out of a node if its value is 1.

Input Format

The first line consists of two integers R and C, denoting the number of rows and columns of the matrix, followed by R rows of C columns of binary digits.

The last two lines contain two integers representing the source node and the destination node, respectively.

Output Format

Print a single integer denoting the shortest path to the destination, or -1 if no path exists.

Note: A breadth-first search (BFS) of an "unweighted" graph (a graph where there is no cost to travel from one vertex to another) will return the shortest path, due to the expanding radial nature of the search. However, if the graph was "weighted" (there *was* a cost to travel from one node to another), a BFS would not (necessarily) find the shortest path. Finding the shortest path of a weighted graph requires ***Dijkstra's algorithm,*** which preferentially chooses edges with lower costs. You'll learn about Dijkstra's algorithm soon!

The input matrix is an adjacency matrix where 1 represents one or more edges exist at a node with adjacent nodes and 0 represents no connection exists.

Hints:

- It may help to make a Location class, such that the queue will store Location objects
- You need some way of tracking which nodes have been visited, such that you don't process nodes more than once.
- The number of steps from one node to the next node is one more than the number of steps it took to reach the previous node.

SAMPLE INPUT

9 10
1 0 **1** **1** **1** 1 0 1 1 1
1 0 **1** 0 **1** 1 1 0 1 1
1 **1** **1** 0 **1** 1 0 1 0 1
0 0 0 0 **1** 0 0 0 0 1
1 1 1 0 1 1 1 0 1 0
1 0 1 1 1 1 0 1 0 0
1 0 0 0 0 0 0 0 0 1
1 0 1 1 1 1 0 1 1 1
1 1 0 0 0 0 1 0 0 1
0 0
3 4

SAMPLE OUTPUT

```
11
```

//the shortest path is bolded above, purely for demonstration purposes