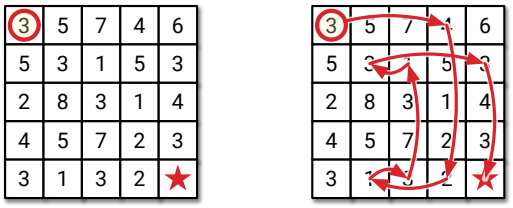
For each of the problems below, transform the input into a graph and apply a standard graph algorithm that you have seen in class. Whenever you use a standard graph algorithm, you ***must*** provide the following information. (I recommend actually using a bulleted list.)

- What are the vertices?
- What are the edges? Are they directed or undirected?
- If the vertices and/or edges have associated values, what are they?
- What problem do you need to solve on this graph?
- What standard algorithm are you using to solve that problem?
- What is the running time of your entire algorithm, *including* the time to build the graph, *as a function of the original input parameters*?

1 A ***number maze*** is an $n \times n$ grid of positive integers. A token starts in the upper left corner; your goal is to move the token to the lower-right corner. On each turn, you are allowed to move the token up, down, left, or right; the distance you may move the token is determined by the number on its current square. For example, if the token is on a square labeled 3, then you may move the token three steps up, three steps down, three steps left, or three steps right. However, you are never allowed to move the token off the edge of the board.



A 5×5 number maze that can be solved in eight moves.

Describe and analyze an efficient algorithm that either returns the minimum number of moves required to solve a given number maze, or correctly reports that the maze has no solution. For example, given the number maze shown above, your algorithm should return the integer 8. Your input is a two-dimensional array $M[1 \,..\, n, 1 \,..\, n]$ of positive integers.

The remaining problems consider variants of problem 1, where the sequence of moves must satisfy certain constraints to be considered a valid solution. For each problem, your goal is to describe and analyze an algorithm that either returns the minimum number of moves in a *valid* solution to a given number maze, or reports correctly that no valid solution exists.

2 Suppose a sequence of moves is considered *valid* if and only if the moves alternate between horizontal and vertical. That is, a valid move sequence never has two horizontal moves in a row or two vertical moves in a row.

Describe and analyze an algorithm that either returns the minimum number of moves in a valid solution to a given number maze, or reports correctly that no valid solution exists.

3 Suppose a sequence of moves is considered *valid* if and only if its length (the number of moves) is a multiple of 5.

Describe and analyze an algorithm that either returns the minimum number of moves in a valid solution to a given number maze, or reports correctly that no valid solution exists.

Harder problems to think about later:

4 Now suppose a sequence of moves is considered *valid* if and only if it does not contain two adjacent moves in opposite directions. In other words, a sequence of moves is valid if and only if it contains no U-turns.

Describe and analyze an algorithm that either returns the minimum number of moves in a valid solution to a given number maze, or reports correctly that no valid solution exists.

5 Now suppose a sequence of moves is considered *valid* if and only if each move in the sequence is longer than the previous move (if any). In other words, a sequence of moves is valid if and only if the sequence of move *lengths* is increasing.

Describe and analyze an algorithm that either returns the minimum number of moves in a valid solution to a given number maze, or reports correctly that no valid solution exists.

6 Finally, suppose a sequence of moves is considered *valid* if and only if the sequence of move lengths is a palindrome. (A palindrome is any sequence that is equal to its reversal.)

Describe and analyze an algorithm that either returns the minimum number of moves in a valid solution to a given number maze, or reports correctly that no valid solution exists.