Areté
DISCOVER. DEVELOP. DELIVER.

Complete the following exercises. You may choose to complete the exercises in C++ or Python. When coding your solutions try to make them as robust as possible and handle any edge cases or possible exceptions that could occur. You will be asked to review and/or demonstrate your solution during the interview. Note that this is not a pass/fail scenario, but rather an opportunity for us to understand how you solve problems. Specifically, we'll be looking at the following:

- Correctness: We want to see that you understood the problem and found a solution that works for all possible inputs.
- Code Quality: We are building systems that should stand the test of time, and as such need to be maintainable. If your solution is difficult to understand, it will be difficult to evaluate and maintain.

You are being provided a template codebase and input files to complete the following problems. In each of the language directories (`cpp`, `python`) you will find a `candidate` file. Please implement your solutions in the file located in the directory of the respective language of your choice. The files that you will need to implement and test your solution can be found in the `inp_files` directory.

1. The provided `stores.csv` file contains a list of stores with name (`name`), x-position (`x`), and y-position (`y`). Your task is to create a function that will return the `n_stores` nearest store names to an input position (`pos`).
    a. In the provided source file, implement your solution to this problem in the `n_nearest_without_obstacles` function.

2. The previous exercise assumes a straight path between two points. Many times, however, there are obstacles that we need to get around to get from point A to point B. In addition to the `stores.csv` file, you have been provided the `stores_map.txt` file. This file represents a grid containing the store locations within its bounds along with some obstacles. (Empty spaces are marked as '.', obstacles are marked as 'X'. For your convenience, there is also a second map file, `stores_map_with_stores.txt`, which includes the redundant information of where stores are located, marked as 'S'). Your task is to create a pathfinder that finds the shortest path between an input position (`pos`) and every store in the given list. Your solution should return the `n_stores` nearest store names, using the path lengths as the distance between the input position and the stores (instead of straight-line distance as in problem 1).
    a. In the provided source file, implement your solution to this problem in the `n_nearest_with_obstacles` function.