Problem 1)
Executing this code in terminal type:
```
python3 Problem1.py input1.txt
```

I struggled a lot with Problem 1, therefore there are 3 python files each containing a method for finding the shortest path. Problem1.py was built using the pseudo code provided. I struggled with understanding how the edge costs factored in, but after spending time on Dijkstra's, I think if I go back, I should be able to finish this algorithm.
The next file is redoP1.py. This was my original implementation, wherein which I recursively find nearby neighbors, and keep expanding from the start point until I get to the end. The flaw for this algorithm was that I didn't take into account the edge costs.
The final file is my Dijkstra's implementation. Just to double check if it would work on a smaller scale prior to going to the larger map.

Problem 2)
Executing this code in terminal type:
```
python3 Problem2.py input_1.txt
```

I started on Problem 2 a bit late after spending too long on the first problem. My Dijkstra's algorithm works, but instead of checking every single vertex, I just look at the neighbors. I didn't want to make the algorithm inefficient by checking every single vertex, but by getting caught up in that, I didn't get my Dijkstra's working perfectly to output the exact output as specified in the output files. But, it does find the shortest path by iteratively searching the neighbors of all the previously visited spots, and travels to the spot with the lowest edge cost.