Problem 2662: Minimum Cost of a Path With Special Roads

Problem Information

Difficulty: Medium
Acceptance Rate: 41.91%
Paid Only: No
Tags: Array, Graph, Heap (Priority Queue), Shortest Path

Problem Description

You are given an array `start` where `start = [startX, startY]` represents your initial position `(startX, startY)` in a 2D space. You are also given the array `target` where `target = [targetX, targetY]` represents your target position `(targetX, targetY)`.

The **cost** of going from a position `(x1, y1)` to any other position in the space `(x2, y2)` is `|x2 - x1| + |y2 - y1|`.

There are also some **special roads**. You are given a 2D array `specialRoads` where `specialRoads[i] = [x1i, y1i, x2i, y2i, costi]` indicates that the `ith` special road goes in **one direction** from `(x1i, y1i)` to `(x2i, y2i)` with a cost equal to `costi`. You can use each special road any number of times.

Return the **minimum** cost required to go from `(startX, startY)` to `(targetX, targetY)`.

Example 1:

Input: start = [1,1], target = [4,5], specialRoads = [[1,2,3,3,2],[3,4,4,5,1]]

Output: 5

Explanation:

1. (1,1) to (1,2) with a cost of |1 - 1| + |2 - 1| = 1. 2. (1,2) to (3,3). Use `specialRoads[0]` with the cost 2. 3. (3,3) to (3,4) with a cost of |3 - 3| + |4 - 3| = 1. 4. (3,4) to (4,5). Use `specialRoads[1]` with the cost 1.

So the total cost is 1 + 2 + 1 + 1 = 5.

Example 2:

Input: start = [3,2], target = [5,7], specialRoads = [[5,7,3,2,1],[3,2,3,4,4],[3,3,5,5,5],[3,4,5,6,6]]

Output: 7

Explanation:

It is optimal not to use any special edges and go directly from the starting to the ending position with a cost |5 - 3| + |7 - 2| = 7.

Note that the `specialRoads[0]` is directed from (5,7) to (3,2).

Example 3:

Input: start = [1,1], target = [10,4], specialRoads = [[4,2,1,1,3],[1,2,7,4,4],[10,3,6,1,2],[6,1,1,2,3]]

Output: 8

Explanation:

1. (1,1) to (1,2) with a cost of |1 - 1| + |2 - 1| = 1. 2. (1,2) to (7,4). Use `specialRoads[1]` with the cost 4. 3. (7,4) to (10,4) with a cost of |10 - 7| + |4 - 4| = 3.

Constraints:

* `start.length == target.length == 2` * `1 <= startX <= targetX <= 105` * `1 <= startY <= targetY <= 105` * `1 <= specialRoads.length <= 200` * `specialRoads[i].length == 5` * `startX <= x1i, x2i <= targetX` * `startY <= y1i, y2i <= targetY` * `1 <= costi <= 105`

Code Snippets

C++:

```
class Solution {
public:
int minimumCost(vector<int>& start, vector<int>& target, vector<vector<int>>&
specialRoads) {

}
};
```

Java:

```
class Solution {
public int minimumCost(int[] start, int[] target, int[][] specialRoads) {

}
}
```

Python3:

```
class Solution:
def minimumCost(self, start: List[int], target: List[int], specialRoads:
List[List[int]]) -> int:
```