Problem 1036: Escape a Large Maze

Problem Information

Difficulty: Hard
Acceptance Rate: 36.03%
Paid Only: No
Tags: Array, Hash Table, Depth-First Search, Breadth-First Search

Problem Description

There is a 1 million by 1 million grid on an XY-plane, and the coordinates of each grid square are `(x, y)`.

We start at the `source = [sx, sy]` square and want to reach the `target = [tx, ty]` square. There is also an array of `blocked` squares, where each `blocked[i] = [xi, yi]` represents a blocked square with coordinates `(xi, yi)`.

Each move, we can walk one square north, east, south, or west if the square is **not** in the array of `blocked` squares. We are also not allowed to walk outside of the grid.

Return `true` _if and only if it is possible to reach the_`target` _square from the_`source` _square through a sequence of valid moves_.

Example 1:

Input: blocked = [[0,1],[1,0]], source = [0,0], target = [0,2] **Output:** false **Explanation:** The target square is inaccessible starting from the source square because we cannot move. We cannot move north or east because those squares are blocked. We cannot move south or west because we cannot go outside of the grid.

Example 2:

Input: blocked = [], source = [0,0], target = [999999,999999] **Output:** true **Explanation:** Because there are no blocked cells, it is possible to reach the target square.

Constraints:

* `0 <= blocked.length <= 200` * `blocked[i].length == 2` * `0 <= xi, yi < 106` * `source.length == target.length == 2` * `0 <= sx, sy, tx, ty < 106` * `source != target` * It is guaranteed that `source` and `target` are not blocked.

Code Snippets

C++:

```
class Solution {
public:
bool isEscapePossible(vector<vector<int>>& blocked, vector<int>& source,
vector<int>& target) {

}
};
```

Java:

```
class Solution {
public boolean isEscapePossible(int[][] blocked, int[] source, int[] target)
{

}
}
```

Python3:

```
class Solution:
def isEscapePossible(self, blocked: List[List[int]], source: List[int],
target: List[int]) -> bool:
```