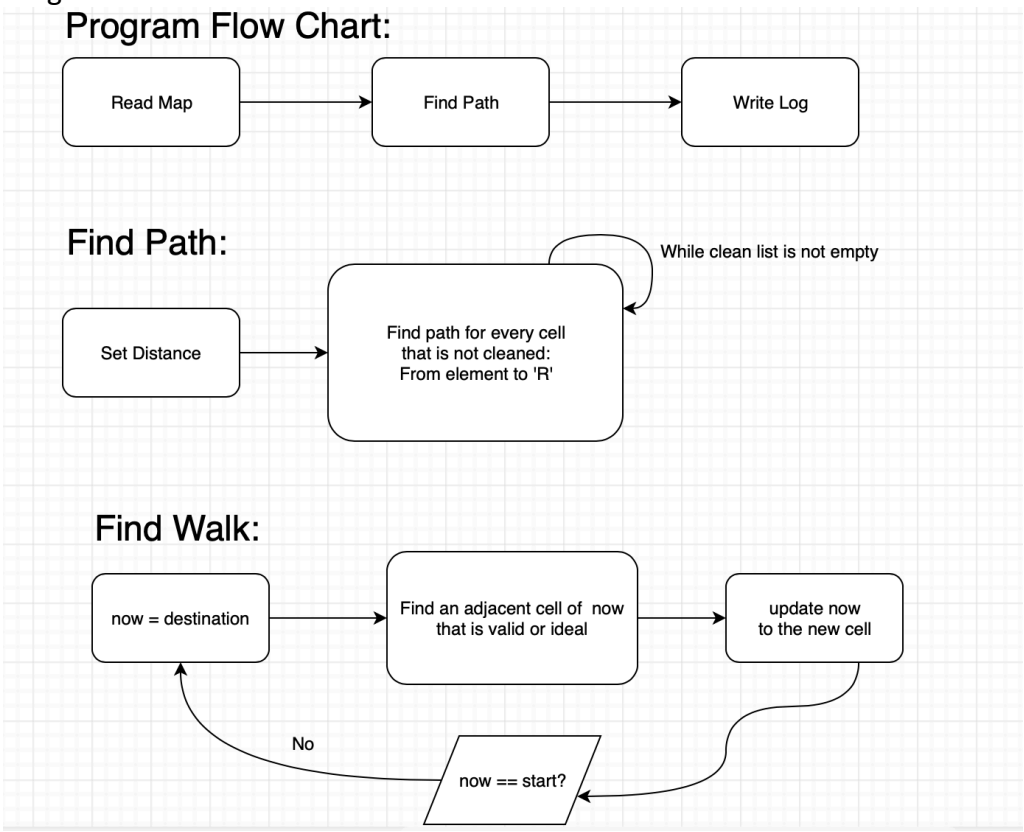
Report

1. Project Description:
    a. Program Flow Chart:



    b. Detailed Description:
        i. Program Flow:
            - ReadMap: Reads the input from file.
            - FindPath: Calculates the answer.
            - WriteLog: output the answer to file.
        ii. Function “FindPath”:
            - SetDist: Calculate the distance from cell ‘R’ to every cell using BFS.
            - “SetDist” simultaneously generates a list of cells to clean.
            - Loop through the to-clean list and call “FindWalk” to find a cycle that goes to every cell and comes back.
        iii. Funciton “FindWalk”:
            - Starting from the destination cell, find a valid or ideal adjacent cell, loop until you get to the start cell.
            - Definition of a valid cell: indices in range, is walkable (!= ‘1’), distance from start cell is 1 step less than current cell.
            - Definition of an ideal cell: a valid cell that is not cleaned.

2. Test case design:

```
15 15 43
0 0 0 0 0 0 1 R 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
```

a. Idea: The idea is to maximize repeated cells walked. Thus, I designed a pattern with several narrow paths and forced the cleaning robot to run to the leftmost and to the rightmost many times.
b. Battery: One charge of battery is carefully calculated such that it is only enough for the robot to get to the farthest cell and travel back.
c. Scale: Above is a miniature version of my testcase, the original one is 1000*1000.