```
 1 //
 2 // Created by mark- on 2023-02-09.
 3 //
 4 #include "Cell.h"
 5 #ifndef ASSIGNMENT_2_NODE_H
 6 #define ASSIGNMENT_2_NODE_H
 7
 8
 9 class Node {
10
11 public:
12     Cell m_data;
13     Node* m_next {nullptr};
14
15 };
16
17
18 #endif //ASSIGNMENT_2_NODE_H
19
```

```
1 //
2 // Created by mark- on 2023-02-09.
3 //
4 #include "Node.h"
5 #include <iostream>
6 #ifndef ASSIGNMENT_2_STACK_H
7 #define ASSIGNMENT_2_STACK_H
8
9
10 class Stack {
11 private:
12     Node* m_first {nullptr};
13
14 public:
15
16     void push(Cell cell);
17     void pop();
18     Cell peek();
19     //void start_maze(std::string *array);
20
21      //~Stack();
22     friend std::ostream &operator<<(std::ostream &output, Stack &stack);
23 };
24
25
26 #endif //ASSIGNMENT_2_STACK_H
27
```

```cpp
//
// Created by mark- on 2023-02-09.
//

#include "Stack.h"
#include "Node.h"
#include <iostream>


void Stack::push(Cell cell) {
    // create a new node
    auto node = new Node();
    node->m_data = std::move(cell);
    node->m_next = m_first;
    m_first = node;
}

void Stack::pop() {
    // remove the first node

    // check for empty stack
    if (m_first == nullptr) {
        // maybe throw exception, print message or do nothing....
        return;
    }
    // one node exists
    auto node = m_first;
    m_first = m_first->m_next;
    delete node;
}

Cell Stack::peek() {
    if (m_first == nullptr) return {0,0};
    return m_first->m_data;
}

//~Stack() {
//    auto node = m_first;
//        while (node != nullptr) {
//            auto temp = node;
//            node = node->m_next;
//            delete temp;
//        }
//    while (m_first != nullptr) {
//        pop();
//    }
//}


std::ostream& operator<<(std::ostream& output, Stack& stack) {
    auto node = stack.m_first;
    while (node != nullptr) {
        output << "Cell X" << node->m_data.x << ", ";
        output << "Cell Y " << node->m_data.y << std::endl;
        node = node->m_next;
    }
    return output;
}
```

```
 1 //
 2 // Created by mark- on 2023-01-22.
 3 //
 4 #include "stack.h"
 5 #ifndef ASSIGNMENT1_READFILE_H
 6 #define ASSIGNMENT1_READFILE_H
 7
 8
 9
10 class ReadFile {
11
12 public:
13     void read_file(std::string inFile, std::string outFile, std::string *array);
14     void print_file(std::string outFile, std::string *array);
15 };
16
17
18 #endif //ASSIGNMENT1_READFILE_H
19
```

```cpp
//
// Created by mark- on 2023-01-22.
//
#include <iostream>

#include <string>

#include <fstream>
#include <string>
#include <iostream>
#include <exception>
#include <cstdlib>
#include "ReadFile.h"
#include "Stack.h"

using namespace std;

void ReadFile::read_file(std::string inFile, std::string outFile, std::string *array) {
    try {
        char character; // declaring string
        ifstream myFileIn; // file in stream reading only
        ofstream myFileOut; //  file out stream writing only
        myFileIn.open("../tests/"+ inFile, ios::in | ios::out);// original txt file
        //char myArray[51][51];
        string line;
        // open for writing
        if (myFileIn.is_open()) {
            cout << "File Open" << endl; // confirmation of successful file open
            int counter = 0;
            while (!myFileIn.eof()) {// continue until end of file
                getline(myFileIn,line);
//                for(int i = 0;i < 51; i++){
                //line = array[counter];
                array[counter] = line;
                counter++;
//                    }

            }

            myFileIn.close(); // closing file in stream

            cout << "File closed" << endl;

        } else {
            cout << "Input file failed to open." << endl;

        }
//
    }
//        catch (MyException& e) {
//            cout << e.error() << endl;
//        }
    catch (exception &e) {
        cout << "Generic error" << endl;
    }
    catch (...) {
        cout << "General error" << endl;
    }
}

void ReadFile::print_file(std::string outFile, std::string *array) {
    std::ofstream myFileOut;
    myFileOut.open("../solved/"+ outFile, std::ios::out);
    for (int i = 0; i < 51; i++) {
        myFileOut << array[i] << endl;
        cout << array[i] << endl;
    }
    myFileOut.close();

}

```

```cpp
//
// Created by mark- on 2023-02-09.
//

#include "Node.h"

```

```
 1 //
 2 // Created by mark- on 2023-02-10.
 3 //
 4 #include "Stack.h"
 5 #include "Node.h"
 6 #ifndef ASSIGNMENT_2_MAZE_H
 7 #define ASSIGNMENT_2_MAZE_H
 8
 9
10 class Maze {
11
12 public:
13     void start_maze(std::string *array, Stack stack);
14     void check_cell(char *array[]);
15 };
16
17
18 #endif //ASSIGNMENT_2_MAZE_H
19
```

```cpp
//
// Created by mark- on 2023-02-10.
//

#include "Maze.h"
#include <iostream>
#include "Node.h"
#include "Stack.h"
#include <vector>

void Maze::start_maze(std::string *array, Stack stack) {
    std::string line;
//    Cell cell_array[51][51];
    //char char_array[51][51];
    std::vector<Cell> vector;
    std::vector<Cell> vector2;
    for (int i = 0; i < 51; i++) {
        for (int j = 0; j < 51; j++) {
            //char_array[i][j] = array[i][j];
//            std::cout << char_array[i][j];
//            if(j == 50){
//                std::cout << std::endl;
//            }
//            check_cell(char_array);

            if (i == 0 & j == 0) {
                stack.push({1, 0, '#'});
                //char_array[1][0] = '#';
                array[1][0] = '#';
                vector2.push_back({0, 0, '#'});
            }


            //check_cell(char_array[]);
//            for (int k = 0; k < 51; k++) {
//            for(array[stack.peek().x][(stack.peek().y)] != '|') {
//                if (i > 0) {
            if (j < 50 && i < 50 && i > 0 && j > 0) {
                if (stack.peek().y < 100) {
                    char right = array[stack.peek().x][stack.peek().y + 1];
                    char down = array[stack.peek().x + 1][stack.peek().y];
                    char left = array[stack.peek().x][stack.peek().y - 1];
                    char up = array[stack.peek().x - 1][stack.peek().y];
                    char test = array[1][1];
                    int x = stack.peek().x;
                    int y = stack.peek().y;
                    if (right != '+' && right != '-' && right != '|' && right != '#') {

                        stack.push({stack.peek().x, stack.peek().y + 1, '#'});
                        array[stack.peek().x][stack.peek().y] = '#';
                        vector.push_back({stack.peek().x, stack.peek().y, '#'});

                    } else if (down != '+' && down != '-' && down != '|' && down != '#') {

                        stack.push({stack.peek().x + 1, stack.peek().y, '#'});
                        array[stack.peek().x][stack.peek().y] = '#';
                        vector.push_back({stack.peek().x, stack.peek().y, '#'});

                    } else if (left != '+' && left != '-' && left != '|' && left != '#') {
                        if (vector2[vector2.size()].x == stack.peek().x &&
                            vector2[vector2.size()].y == stack.peek().y) {
                            bool thing = true;
                        }

                        stack.push({stack.peek().x, stack.peek().y - 1, '#'});
                        array[stack.peek().x][stack.peek().y] = '#';
                        vector.push_back({stack.peek().x, stack.peek().y, '#'});

                    } else if (up != '+' && up != '-' && up != '|' && up != '#') {

                        stack.push({stack.peek().x - 1, stack.peek().y, '#'});
                        array[stack.peek().x][stack.peek().y] = '#';
                        vector.push_back({stack.peek().x, stack.peek().y, '#'});
```

```
74
75                      } else {
76
77                          vector2.push_back({stack.peek().x, stack.peek().y, '#'});
78
79                          for (int k = 0; k < vector2.size(); k++) {
80                              if (stack.peek().y == vector2[k].y && stack.peek().x == vector2[k].x) {
81
82                                  array[stack.peek().x][stack.peek().y] = '-';
83                                  stack.pop();
84                                  //vector2.push_back({stack.peek().x, stack.peek().y, '#'});
85
86                              }
87                          }
88                          while (!vector2.empty()) {
89                              vector2.pop_back();
90                          }
91
92                      }
93                  }
94              }
95
96
97
98
99  //
100 //                  }
101 //              }
102 //              if(char_array[i][j] == ' '){
103 //                  stack.push({i,j,'#'});
104 //                  char_array[i][j] = '#';
105 //              }
106
107 //                  if (char_array[(stack.peek().x) + 2][(stack.peek().y) + 2] == '+' || char_array[i][j] == '-' ||
108 //                      char_array[i][j] == '|') {
109 //
110 //                  }
111
112 //              }
113         }
114     }
115 //    for (int i = 0; i < 51; i++) {
116 //        for (int j = 0; j < 51; j++) {
117 //            array[i][j] = char_array[i][j];
118 //        }
119 //    }
120
121
122 }
123
124 //void Maze::check_cell(char *array[]) {
125 //
126 //}
127
128
```

```cpp
#include <iostream>
#include "Stack.h"
#include "ReadFile.h"
#include "Maze.h"

using namespace std;

int main(int argc, char *argv[]) {
    if (argc == 3) {
        Stack Stack;
        ReadFile readFile;
        Maze maze;
        string myArray[51];
        readFile.read_file(argv[1], argv[2], myArray);
        maze.start_maze(myArray, Stack);

        readFile.print_file(argv[2], myArray);
//        for (int i = 0; i < 51; i++) {
//            cout << myArray[i] << endl;
//        }
    }


//        linkedList =  ReadFile::readfile(argv[1],linkedList);
//        linkedList = textEditor.startTextEditor(linkedList);
//        cout << linkedList << endl;

    else{
        cout << "Check Command Line Arguments" << endl;
    }
    return 0;
}
```

```
//
// Created by mark- on 2023-02-09.
//

#ifndef ASSIGNMENT_2_CELL_H
#define ASSIGNMENT_2_CELL_H


class Cell {

public:
    int x;

    int getX() const;

    void setX(int x);

    int getY() const;

    void setY(int y);

    int y;
    char symbol = '#';
};


#endif //ASSIGNMENT_2_CELL_H

```

```cpp
//
// Created by mark- on 2023-02-09.
//

#include "Cell.h"

int Cell::getX() const {
    return x;
}

void Cell::setX(int x) {
    Cell::x = x;
}

int Cell::getY() const {
    return y;
}

void Cell::setY(int y) {
    Cell::y = y;
}

```