```cpp
//////////////////////////////////////////////////////////////////////
//                                                                  //
// Author       : Gregorio Lozada                                   //
// Date         : 4/5/2018                                          //
//                                                                  //
// Homework Assignment 3:   Maze                                    //
//                                                                  //
// This program solves an 8 x 13 maze.                              //
//                                                                  //
// Output       : Maze displaying the solved patth and dead ends.   //
//                                                                  //
//////////////////////////////////////////////////////////////////////

#include <stdio.h>
#include "Stack.h"

void MovePoint(Stack *stack, Point *runner, int row, int column);

int main() {
    //Variables
    char maze[10][16] =
    {   {"000000000000000"},
        {"0 0 0 000 0  00"},
        {"0             0"},
        {"0 0 0000000 0 0"},
        {"0 0   0   0   0"},
        {"0 000  00  0000"},
        {"0    0 0      0"},
        {"00 0 0 0 000 00"},
        {"00   0   0    0"},
        {"000000000000000"}
    };

    int mazeSize = (sizeof(maze) / sizeof(maze[0][0]));

    Stack stack(mazeSize);

    Point runner(1, 1);
    Point end(8, 13);

    //WHILE runner is not at the end of maze and stack is not empty
    do {
        /*
        IF there's an empty space above, below, or next to runner
            Change maze character to '*'
            MovePoint
        ELSE
            Change maze character to 'D'
            Set runner point to popped point in stack
        */
        if (maze[runner.row][runner.column + 1] == ' ') {
            maze[runner.row][runner.column] = '*';
            MovePoint(&stack, &runner, runner.row, runner.column + 1);
        }
        else if (maze[runner.row - 1][runner.column] == ' ') {
            maze[runner.row][runner.column] = '*';
            MovePoint(&stack, &runner, runner.row - 1, runner.column);
        }

        else if (maze[runner.row][runner.column - 1] == ' ') {
            maze[runner.row][runner.column] = '*';
            MovePoint(&stack, &runner, runner.row, runner.column - 1);
        }
        else if (maze[runner.row + 1][runner.column] == ' ') {
            maze[runner.row][runner.column] = '*';
            MovePoint(&stack, &runner, runner.row + 1, runner.column);
        }
        else {
            maze[runner.row][runner.column] = 'D';
            runner.Set(stack.Pop());
        }
    }
    while ((runner.row != end.row || runner.column != end.column)
            && stack.Size() != 0);

    /*
    IF runner made it our of the maze successfully
        PRINT congratulations message
    ELSE
        PRINT "Cannot solve maze."
    */
    if (stack.Size() != 0) {
        maze[end.row][end.column] = '*';

        printf("Congratulations!\n");
        printf("You made it out of the maze!\n");
    }
    else {
        printf("Cannot solve maze.\n");
    }

    printf("\n");

    //PRINT maze with path
    for (int i = 0; i < (sizeof(maze) / sizeof(maze[0])); i++) {
        for (int j = 0; j < (sizeof(maze[0]) / sizeof(maze[0][0])); j++) {
            printf("%c", maze[i][j]);
        }
        printf("\n");
    }

    printf("\n");

    return 0;
}

void MovePoint(Stack *stack, Point *runner, int row, int column) {
    //PUSH runner point to stack
    stack->Push(Point(runner->row, runner->column));

    //SET runner point to empty point
    runner->Set(row, column);
}
```