HACETTEPE UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING

BBM203

SOFTWARE LABORATORY I

ASSIGNMENT – 1

Student's Name: Onur Tolga Kesemen

Student's Number: 21228511

Due Date: 22.10.2017

Advisor: R.A Gültekin Işık, Dr. Burcu Can, Dr. Sevil Şen, Dr. Adnan Özsoy

Software Usage

This program takes one input file. In the file there should be a maze like below. Program takes it and looks for a path. If it can find then writes the path to a file named “path.txt”, otherwise writes an error message to “path.txt”.

Sample Input

1111S11110

0000010001

110100010d

t001111110

0100000001

0111111101

1111111101

00000D01T1

0111110001

0000E01110

Sample Output

Start S E E E E S S S S E E S S W W S S S g N N N E E N N W W N N N N W W W W S S S E E N a S W W N N N E E E E S S S S E E S S E S S W S S E S S S A W W N b N W W S W S Exit

Sample Error Message

ERROR!!!! No path found!

Software Design Notes

Problem

In this experiment the problem is finding a solution path in a given maze which includes 0 (for clear way), 1 (for walls), lower case (for keys), upper case (for doors). In the maze, solution path should starts from ‘S’ which is in the first row and ends at ‘E’ which is in the last row. Solution path does not have to be the shortest one. It shouldn’t contain false paths but if path meets with a key then it is a right path even if it can not go from there to anywhere.

Solution

In my algorithm, i took the maze in a 2d array of structure. This structure includes data whose type is char and marked whose type is int. I used the variable marked for already passed cells in the maze. If it is already passed then marked equals 1, otherwise it is 0.

There is a function called ‘mazeRunner’ and it is a recursive function which calls itself if needed. This function takes coordinates of current position in the maze and looks around to find a way which can go. It looks four way respectively: East, South, West, North. If one is not passed yet then function looks what is the data in it.

I took the path in a dynamic stack. If the path is clear program pushes it to stack otherwise pops it from stack and changes the size of stack dynamically. And there is functions like push, pop, isfull… for stack control.

At last, i write a function called ‘unmarkAll’. Because, i needed to go forward after getting a key but no solution path from there. The program behaves that key as starting point but do not erase the way which is used when coming that key, clear all marks in the array and keep looking for solution path.