Alican Özeloğlu - 66277019558

EEM480 ALGORITHMS AND COMPLEXITY

HOMEWORK 3

TASK

The purpose of the homework is to find close paths in a given maze.

a	r	h	g	f	e	t	g	k	l
b	j	i	r	b	d	l	y	n	m
c	k	t	a	b	c	g	v	d	a
k	l	m	n	h	a	z	m	c	b
a	f	e	c	g	b	c	d	f	e
n	d	b	r	f	e	d	h	v	s
x	d	k	a	y	f	m	n	n	n
z	s	w	b	v	b	l	b	a	p
h	a	g	c	r	j	k	p	w	l
b	k	l	n	g	s	d	k	z	d

Figure 1

Maze will have a structure as shown in figure 1. The task of our algorithm is to find loops that match the sequence of the English alphabets in the maze.

HOW IT'S DONE

The Project is created in the Apache Netbeans IDE 11.1 environment, an interface named "MazeInterface" was created and the class was filled with the data given in the assignment. Then a class named "MazeClass" was created and the methods in the MazeInterface were overridden in the MazeClass. A new class named "NodeForCoordinate" has been created that holds rows, columns, and address information.

Then a class named "StackForCoordinate" was created. In this class, the algorithm of push(), pop(), top(), isEmpty() methods was created. An object of type NodeForCoordinate was created for using in the push () method. If the stack size is 0, the object was created using the first constructor. If the stack size is not 0, the object is created using the second constructor. Stack size is increased by 1 each time a new node is added. In pop () method, stack size was checked. If the stack size is not 0, the last added node was deleted. When the last added node is deleted, the stack size is reduced by 1. The row and col information of the last deleted node was returned. In the top () method, the stack size was checked. If the stack size is not 0, the row and col of the last added node are returned. In the isEmpty() method, the stack size was checked. "True" is returned if the stack size is 0.

Then, the "Position" class was created, and this class was allowed to hold the row and column information.

In the MazeClass class, a string of character consisting of the English alphabet "transform" was created and two 2-dimensional arrays of char were created.

InputMaze(String FileName) Method:

The maze size and the content of the maze in the txt file were obtained by using data extraction algorithms from the txt file in the literature. If we call (m)x(n) the size of the maze from the file, a new maze of char type and (m+2)x(n+2) size is created. Using the for loop, content in the old maze was transferred to the new maze, leaving the outermost circumference of the new maze free. Again with

the help of a for loop, the 'a' character was placed around the outermost part of the new maze. So padding was done.

FindLoops() Method:

Two objects named pos and ref were created. Then an object named s0 was created in StackForCoordinate type. For each character in the maze, it is checked if there is any way around it. Is the index in the transform array of the character on the right 1 more than the index of the character in the current position? This inquiry is done for every direction. If there is a way to go, it goes that way and the row and col values on the path are pushed to the stack. If there's no way to go, it's looking for a reference point (starting point). If there is a starting point, we know that there is a loop, the elements in the stack are saved in an array because we found the loop path. The algorithm will first look at nearby roads for the current position. if there is no way, it will check to see if there is a starting point around. If there is no starting point, this may mean that we have entered the wrong path. we need to go back and check if there is a more accurate route along the same route. The algorithm takes the last position from the stack that holds the path information and goes back. But it is necessary to take prevention to avoid entering that road again. An array of objects named "dontgo" was created. This object saved the wrong path positions and ensures that we should never enter that road again. Because algorithm control the “dontgo” positions before make decision about choosing way.

FindLoops(String FileName):

This method follows the same procedure with the FindLoops() method. The only differences is about showing results. This method is writing the result in a file.