Gebze Technical University
Computer Engineering

CSE 222 - 2018 Spring

HOMEWORK 3 REPORT

AHMET DENİZLİ
161044020

Course Assistant: Özgü Göksu

1 INTRODUCTION

1.1 Problem Definition

In first part, there is a binary digital image is represented through a matrix of integers each element of which is either 1(white) or 0 (black). Components of white ones are asked to be found and shown.

In second part, there is a infix expression which can contain variables, values, functions and parenthesis. The infix expression's value asked to be found and shown.

1.2 System Requirements

Any release of JDK must be installed to compile this program. The compiled version of program will work any jvm installed environment.
Program will work at least 64 MB of physical RAM and 98 MB of free disk space for jvm.

2 METHOD

2.1 Use Case Diagrams

In first part compile file with this command "javac part1.java" from command line. Then run program with this command "java part1 filename" from command line which filename is text file name containing a binary digital image represented.
In second part compile file with this command "javac part2.java" from command line. Then run program with this command "java part2 filename" from command line which filename is text file name containing a infix expression.

2.2 Problem Solution Approach

In first part looks whether cells are 1, if one cell is 1 then pushes it to stack and looks to the right, left, bottom, top of the cell. Then pop first cell from the stack and pushes cells around of this cell if they are 1. Looks all cells around of the first cell iteratively in same way until the stack is empty and shows them with same letter. Using Depth-first search algorithm for effective use to stack. Time complexity for part is O(N), where is N total number of elements.

In second part, there is two stack for postfix expression. First stack is using for storing operands and second stack is using for storing operators. The algorithm of part2 is:
Scans the infix expression from left to right.
If the scanned character is an operand, push it to operands stack.
Else
…..If the precedence of the scanned operator is greater than the precedence of the operator in the stack(or the stack is empty or the stack contains a ‘(‘), push it.
…..Else, Pop all the operators from the operators stack which are greater than or equal to in precedence than that of the scanned operator, and calculate operands from operators stack with this operatör and push result to operators stack. After doing that Push the scanned operator to the operators stack.
If the scanned character is an ‘(‘, push it to the stack.
If the scanned character is an ‘)’, pop the stack and output it until a ‘(‘ is encountered, and discard both the parenthesis. Then calculates operands from operands stack in order with operators.
Repeat this steps until infix expression is scanned.

3 RESULT

3.1 Test Cases

Tests made with error conditions, no data, little data and big data sizes. All errors handled with exceptions. Data size doesn’t effect anything but the working time of functions which we want to calculate and compare to analyze their performances

3.2 Running Results

Running results as intellij terminal outputs can be found on folder Outputs.