Gebze Technical University
Computer Engineering

CSE 222 - 2018 Spring

HOMEWORK 3 REPORT

MELİH YABAŞ
161044072

Course Assistant: ÖZGÜ GÖKSU

1 INTRODUCTION

1.1 Problem Definition

Part1:

If I try to define the problem basically, I tell that there are some islands on the sea and we must calculate how many islands exist.

1 represents a piece of island and 0 represents the sea.

If these 1 chararecters connected each other they are an island.

These connection through their top, left, right or bottom. (Not cross.)

Part2:

Basically, we need to write a calculator. This calculator takes all arguments from a txt file. Additionally, there are some variables and their values. We need to use these varible values to calculate the operations correctly.

These calculator sensitives parantheses, sin, cos, abs and basic operators (+,-,*,/).

1.2 System Requirements

Part1:
Needful Memory: 152kb
Computers and smartphones of today can work this program.
It needs a JVM(OpenJDK 64 Bit Server recommendated)
All common operating systems can work it.

Part2:
Needful Memory: 248kb
Computers and smartphones of today can work this program.
It needs a JVM. (OpenJDK 64 Bit Server recommendated)
All common operating systems can work it.

1.3 Problem Solution Approach

Part1:
Firstly, I thought there is no logical equations to calculate it. And then I decide to create a recursive logic using stack. Stacks are very useful for recursion.
I search all element until I find a '1'. Then I keep its location in the stack.
After I change it, I control all the neighboor of this character. Go there and I push their coordinates too.
After every push operation I control the neighboors and go their. If there is no neighboor, this is an island!
So, I checked all the islands using this method.

Time Complexity:
n = size of our string.

n times nested for loop
c is a real number
n/c is a while loop

time complexity is n^2.

Part2:
We want to calculate some operation that are in a txt file. So, first of all I take this file in the program.
We said that there are some variables and their values. We do not know how many variable exist. So, we need to keep all the variables in a structure. To do this, I create an ArrayList structure and write there 2 array. (One for variables, the other on for their values)
We need to convert this operations infix to postfix. General way to do that is using stack.
Then, parantheses and some operations have priority. So, I use a stack to control it.
If the element is a number, I add it to String. If it is an operator, I push it on the stack.
If the element is a left parantheses, we need to pop the stack element until the top element is right parantheses.
Finally we can pop all the elements and add to the string.
Now, we have some postfix operations.
After these point we must push the numbers to the stack until we see an operator. If we see an operator we pop how many numbers needed and calculate the operation. After that, we must push the result to the stack.
After all this, the last element in the stack is our result.

Time Complexity:

n = size of our string

n for the while loop

n fort he other nested loops

time complexity is n^2

2 RESULT

2.1 Test Cases

Part1:

First, check the test file and it was correctly working. After that, I randomly generate some matrix that have different size and different elements. Some files are really important.

What were they?

1 column matrix, 1 line matrix, matrix that have many islands etc.

After all that I decide this program actually works.

Part2:

First, check the test files and it was correctly working. After that, I generate some different operations.
I increase variable numbers and it was working.
After all that I decide this program actually works.

2.2 Running Results

Part1:

```
0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0
0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1
0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1
0 0 0 0 1 1 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0
0 0 0 0 0 1 1 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1
```

```
There are 9 island.

Process finished with exit code 0
```

```
0 1 1 1 0
1 1 1 1 1
0 1 1 1 0
0 0 0 0 0
1 1 1 1 1
0 0 0 1 0
0 1 1 1 1
1 1 0 0 0
1 0 0 0 1
```

```
"C:\Program Files\Java\jdk-9.0.1\bin\java

There are 3 island.

Process finished with exit code 0
```

Part2:

```
y=3
z=16

( y + sin( y * z ) ) + ( z * abs( -10.3 ) )

Postfix:  3.0 3.0 16.0 * sin + 16.0 -10.3 abs * +

Result: 168.54314470836832
```

```
a=-3
b=1.9
c=9.2
d=-77.7

( b * abs( d ) ) + ( a + cos( sin( b / c ) ) )

Postfix:  1.9 -77.7 abs * abs -3.0 1.9 9.2 / abs sin cos + +

Result: 145.629943689454
```

- Main titles -> 16pt , 2 line break
- Subtitles -> 14pt, 1.5 line break
- Paragraph -> 12pt, 1.5 line break