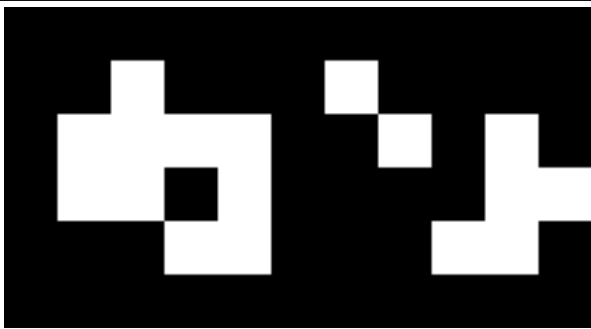


GTU Computer Science and Engineering
CSE 222/505 – Spring 2019
Homework #03
Due Date: 18.03.2019 05:55

1) Counting components.

A binary digital image is represented through a matrix of integers, each element of which is either 1 (white) or 0 (black) **(40 points)**. Example of an image (6 rows, 11 columns) (left) and the corresponding binary image (right):

0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0
0	1	1	1	1	0	0	1	0	1	0	0
0	1	1	0	1	0	0	0	0	1	1	0
0	0	0	1	1	0	0	0	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0



Input

You are to write a program that admits as a commandline argument the path to a ASCII text file containing a binary digital image represented through a matrix of space separated integers such as the one above.

Output

Your program will calculate the number of “white components” and print that number on screen.

Explanation

A white component is a set of white matrix locations, such that, between any two of them, there is a path of white matrix locations, where every consecutive pair are adjacent (through their top, left, right or bottom neighbor). Hence the above example contains four components, shown here with letters A, B, C and D:

```
0 0 0 0 0 0 0 0 0 0 0 0
0 0 A 0 0 0 B 0 0 0 0
0 A A A A 0 0 C 0 D 0
0 A A 0 A 0 0 0 0 D D
0 0 0 A A 0 0 0 D D 0
0 0 0 0 0 0 0 0 0 0 0
```

Assume location (0,0) corresponds to the top left element.

B contains the locations (in column,row format): (6,1)

C contains the locations (in column,row format): (7,2)

D contains the locations (in column,row format): (9,2)(9,3)(9,4)(10,3)(8,4)

etc

e.g. locations (10,3) and (8,4) (in pink) belong to the same white component since there is a path (i.e. a sequence) of white locations starting from (10,3) and ending at (8,4): (10,3),(9,3),(9,4),(8,4)

such that every consecutive location pair $\mathbf{z}=(x_1,y_1)$ and $\mathbf{w}=(x_2,y_2)$ are neighbors (i.e. \mathbf{z} is either on top, bottom, left or right of \mathbf{w} ; e.g. (8,4) is at the left of (9,4)).

You need to visit every matrix element at least once in order to accomplish this goal. So your complexity will be at least $O(n)$ where n is the number of matrix elements. Make sure your program's complexity is no worse than $O(n)$ at the worst case.

Rules

- Don't use recursive functions.
- Your program will be tested with an arbitrary matrix
- Don't use any Java class implementing the Collection interface; (implement your own data structures).
- Hint 1: you can represent your matrix as a 2 dimensional array of integers.
- Hint 2: you might want to use a stack in your algorithm for white component counting.

2) A calculator

Write a program that evaluates an infix expression (**40 points**). You already know that you must first convert it to postfix and then evaluate the postfix expression with a stack based algorithm.

The trick is that the infix expression can contain:

- variables
- functions (only sin, cos, abs)
- parenthesis

Input

The input will be provided through the commandline as a path to file containing a single infix expression (an example is provided at the moodle page: `infix_test_file_1.txt`). The expression's tokens will be space separated.

Output

The infix expression's value.

Rules

- Don't use any Java classes implementing Java's Collection interface; (implement your own data structures).
- Do not call any method from the Math class of java, implement your own sin, cos, abs methods.

PS:

- If you have any questions about the hw, please send an email to ogoksu@gtu.edu.tr,
- Submit your homework as studentID.zip and includes the following files:
 - o Project file,
 - o Report.pdf
- The implementation will be worth 80 points and the report 20 points.
- The submitted homework must be entirely in English.
- Programs with compilation errors will not be accepted.
- No plagiarism!

Good Luck!