Indian Statistical Institute

Semester-I 2021-2022

M.Tech.(CS) - First Year

Lab Test 1 (09 December, 2021)

Subject: Computing Laboratory

Total: 60 marks Duration: 3 hrs.

SUBMISSION INSTRUCTIONS

1. Naming convention for your programs: `cs21xx-test1-progy.c` (assuming `cs21xx` denotes your roll number and `progy` denotes the program number).
2. To submit the solution files (`.c` or `.h`), ensure that they not password protected and mail them together to `<assignisik@gmail.com>` with the subject line as follows: `MTech (CS) 2021-23 cs21xx Computing Lab - labtest1`.
3. You may consult or use slides / programs provided to you as course material, or programs that you have written yourself as part of classwork / homework for this course, but please **do not** consult or use material from other Internet sources, your classmates, or anyone else.
4. Please make sure that your programs adhere strictly to the specified input and output format. **You may lose marks if your program violates the input and output requirements.**
5. Submissions from different students having significant match will be **debarred from evaluation**.

1. Suppose a number of cubical bricks of equal sizes are set up in succession in a row. In some of the positions in the said row, there can be multiple bricks perfectly aligned over each other (as shown in the figure). There are additional bricks of the same shape to continue the construction. However, due to shortage of bricks for further construction, you have been asked to fill some space with quick-setting cement (to be poured onto that row) so that no cement spills over. Your only goal is to save maximum number of cubical bricks by pouring quick-setting cement. The rest of the things will be managed by the construction team. Write a program that takes the number of bricks already set up on the row and return the number of maximum bricks you can save by pouring quick-setting cement.

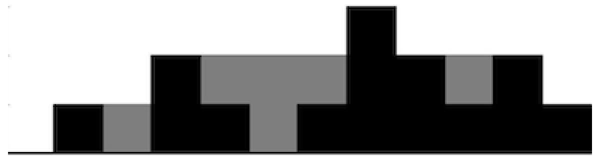

Figure 1: The black cubes are bricks and the gray regions are covered by quick-setting cement.

[20]

Input Format

The input (to be read from stdin) comprises a series of non-negative integers representing the number of bricks aligned over each other as they successively appear in the row.

Output Format

The output (to be printed to stdout) shows the maximum number of bricks that can be saved by pouring quick-setting cement.

Sample Input 0:

```
0 1 0 2 1 0 1 3 2 1 2 1
```

Sample Output 0:

```
6
```

Sample Input 1:

```
4 2 0 3 2 5
```

Sample Output 1:

```
9
```

Sample Input 2:

```
9 0 9 0 9 0 9
```

Sample Output 2:

```
27
```

2. Let the *layerwise rotation* of a matrix denotes the shifting of its elements in a particular layer by a fixed number. A layer represents the set of elements present in rows and columns end-to-end connected. For example, the elements in the i^{th} and $(m-i)^{th}$ rows and i^{th} and $(n-i)^{th}$ columns of an $m \times n$ matrix are in layer i, where $i = \lfloor \min(m,n)/2 \rfloor$. Consider that the counting of row and column number starts from 1. Write a program that will take an input matrix, and the three arguments, namely the rotation type (clockwise or anticlockwise), the number of positions of rotation, and the layer i, and return the in-place rotated matrix. Note that, the input matrix is to be converted to the output matrix without using a third matrix.

   As an example, after layerwise rotation on the matrix by 3 positions anticlockwise at the layer 2 gets converted to the matrix on the right.

$$\begin{matrix} 9 & 0 & 4 & 3 \\ 1 & 4 & 7 & 2 \\ 0 & 0 & 0 & 0 \\ 6 & 9 & 3 & 2 \\ 8 & 3 & 1 & 6 \end{matrix} \quad \longrightarrow \quad \begin{matrix} 9 & 0 & 4 & 3 \\ 1 & 3 & 9 & 2 \\ 0 & 0 & 0 & 0 \\ 6 & 7 & 4 & 2 \\ 8 & 3 & 1 & 6 \end{matrix}$$

[20]

Input Format
The input (to be read from stdin) comprises multiple lines of which the first one takes the dimensions (number of rows followed by number of columns) of the matrix. The next few lines consist of the elements of the matrix in row-major traversal form. The matrix elements in a line are separated by spaces. The last line comprises a letter ‘C’ (denoting clockwise rotation) or ‘A’ (denoting anticlockwise rotation), followed by two numbers, representing the number of positions of rotation, and the layer at which the rotation will happen.

Output Format
The output (to be printed to stdout) shows the layerwise rotated matrix.

Sample Input 0:

```
5 4
9 0 4 3
1 4 7 2
0 0 0 0
6 9 3 2
8 3 1 6
C 2 1
```

Sample Output 0:

```
0 1 9 0
6 4 7 4
8 0 0 3
3 9 3 2
1 6 2 0
```

Sample Input 1:

```
3 3
3 6 4
4 7 2
0 0 1
A 8 1
```

Sample Output 1:

```
3 6 4
4 7 2
0 0 1
```

Sample Input 2:

```
2 2
8 8
9 9
C 1 1
```

Sample Output 2:

```
9 8
9 8
```

3. Write a program that can take a C statement, which uses ternary operator, as user input and return whether the input statement is valid or not. Consider that validity of a statement only depends on the structure of the ternary operator used in the statement.

[20]

Input Format
The input (to be read from stdin) is the C statement.

Output Format
The output (to be printed to stdout) will be `VALID` or `INVALID` depending upon whether the C statement is valid or not, respectively.

Sample Input 0:

```
c = (a < b) ? a : b;
```

Sample Output 0:

```
VALID
```

Sample Input 1:

```
a = 2 > 3 ? 2 : 3 > 4 ? 3 : 4;
```

Sample Output 1:

```
VALID
```

Sample Input 2:

```
val = a ? b: c ? d ? : ;
```

Sample Output 2:

```
INVALID
```