Data Structures and Algorithms (CS F211)
Second Semester 2018-2019
Lab Sheet 8

1. You joined an analytics company as an intern. Since you were new you were given an easy task to start with. Every day you will be given a number and you have to calculate the medians of all the numbers you have received till date. You want to automate the process and you have to complete the task in less than O(n^2) time.

Input:
n (total number of days)
n integers (separated by space)

Output:
n integers each on a single line denoting the median of all the numbers given to you till the i^{th} date. Print them in float format.

Example:
6
12 4 5 3 8 7
Answer:
12.0
8.0
5.0
4.5
5.0
6.0

2. Assume that you are managing a pizza store and you have to schedule different pizza requests. Let us consider that i^{th} request comes at time t_i and take s_i seconds to be processed.
For example, let us consider that there are three requests that arrive at $t_1 = 0$, $t_2 = 1$, $t_3 = 2$ with processing time $s_1 = 3$, $s_2 = 9$, $s_3 = 6$. Now, if you apply first come-first serve rule, then request 1 executes first finishing at $t = 3$, then request 2 which finishes at $t = (3 + 9) = 12$ and then request 3 which finishes at $t = (12 + 6) = 18$. Remember that when one process is running it cannot be interrupted by any others. Now we define waiting time for each process as:
Waiting Time = (Time at which the task is done) – (Time at which the task arrived).
So, waiting time of three processes are 3, 11, & 16 respectively. The average waiting time in this case is (3 + 11 + 16) / 3 = 10. This is not an optimal solution. After serving the first customer at time $t = 3$, you can choose to serve the third customer. In that case, the waiting time will be 3, 7, & 17 respectively. Hence the average waiting time is (3 + 7 + 17) / 3 = 9. Given t_i's and s_i's, your goal is to find the minimum average waiting time.

Input:
In the first line, you will be given an integer ***n*** (number of requests).
In the next ***n*** lines, two space-separated integers will be given. The first is arrival time and second is processing time of each request.

Output:
Minimum average waiting time.

3. Write a program to check for balanced parentheses in an expression. For example, your code should return true for "[()]{}{[()()]()}" while false for "{(]}". The expression may contain other characters as well (other than parentheses).

Input:
A string containing the expression.

Output:
True if its balanced, else false.

4. You love cookies. You have ***n*** cookies with you. Let us assume that the i^{th} cookie has ***A[i]*** sweetness value. You want the sweetness values of all your cookies to be greater than or equal to value ***K***. To do this, you repeatedly mix two cookies with the least sweetness values. You create a special combined cookie with:
Sweetness value = (Least sweetness value + 2*second least sweetness value).
You repeat this procedure until all the cookies in your collection have a sweetness greater than or equal to ***K***. Calculate the number of operations required.

Input:
The first line consists of integers ***n*** - the number of cookies and ***K*** - the minimum required sweetness, separated by a space.
Each of the next lines contains ***n*** integers describing the array ***A*** where ***A[i]*** is the sweetness of the cookie.

Output:
Print exactly one integer — the number of operations needed.

Example:
Input:
6 7
1 2 3 9 10 12

Output:
2

5. Given an array of digits (values are from 0 to 9), find the minimum possible sum of two positive numbers formed from digits of the array. All digits of given array must be used to form the two numbers.

Input:
A single containing the digits separated by space.

Output:
The sum.

Example:
Input:
6 8 4 5 2 3

Output:
604

Explanation:
The minimum sum is formed by numbers 358 and 246.

6. Design a stack that supports push, pop, and retrieving the minimum element in constant time. Make it a menu-driven program where the 3 operations can be done any number of times. The user will be prompted to enter the type of operation and the associated data value.

7. Naruto was once interested in sum of entire array. However, now he is obsessed with minimum element in the array. But in order to make things more interesting he decided to find the minimum element of every sub array and then sum it. Help Naruto find the sum.

Input:
The first line contains the size ***N*** of the array.
The second line contains the ***N*** integers separated by space.

Output:
The sum.

Example:
Input:
4
3 1 2 4

Output:
17

Explanation:
The subarrays are {3}, {1}, {2}, {4}, {3, 1}, {1, 2}, {2, 4}, {3, 1, 2}, {1, 2, 4} and {3, 1, 2, 4}. The sum = 3 + 1 + 2 + 4 + 1 + 1 + 2 + 1 + 1 + 1 = 17.

Input:
4
1 2 3 4

Output
20

8. The city of Matrix is tricky. You are given a ***n*n*** binary matrix map of the city. You will get life points if the submatrix you decide to work on has a single 1. Given the map find the number of submatrices that will give you life points. The submatrices are not essentially square matrices.

Input:
The first line contains the value of ***n***.
Next, the matrix is entered row-by-row, each row input in a new line.

Output:
The number of submatrices.

Example:
Input
3
0 0 0
0 0 0
0 0 0

Output
0

Input
3
0 0 0
0 1 0
0 0 0

Output
16

Explanation:
The submatrices are:

1, 0 1, 1 0, 0 1’, 1 0’, 0 1 0, 0 1 0, 0 0 1’, 0 1 0’, 0 0 0’, 0 0 1’, 0 1 0’, 0 0 0’, 0 0 1’, 0 1 0’, 0 0 0’, 0 0 1, 0 1 0, 0 0 0

0 0 0
0 1 0
0 0 0

9. Hawk Eye is a sharp shooter. He has been training for years before joining as Shield Agent. He has an ability that helps him to aim long distance targets with pinpoint accuracy. Assuming he is currently standing on Stark tower and trying to aim at Thanos whose is standing on the last building in that row. He can only shoot him if the building heights are arranged in a non-increasing order. Thor can help Hawk Eye by performing 2 operations, either cut the building by one floor or add some junk on the building to increase its height by one floor. Find the minimum operations needed by Thor.

Input:
The first line contains the number of towers.
The second line contains the heights of the towers separated by spaces.

Output:
The number of operations required.

Example:
Input:
4
3 1 2 1

Output:
1

Input:
4
3 1 5 1

Output:
4

Input:
4
1 5 5 5

Output:
4

10. Zolo is stuck in a traffic due to dysfunctional traffic light. Zolo is a professional hacker and he can get into the system and change the state of the light. His planet has different types of traffic lights such that there are **N bulbs** on the traffic board and only when all of them are green(**G**) the cars can pass. There are **2** other states also which the bulb can show - Red(**R) &** Yellow(**Y**). Note that the lights are designed such that they follow a state change cyclic pattern as follows:

R------>Y------>G------->R

Once Zolo gets into the system he can select any position ***i*** and update all elements between ***i*** **to min(N,** ***i*** **+ K - 1)** by increasing their state by 1.This whole process takes **1 sec** and he can repeat this process any number of times until he gets all lights = **G .** Find the minimum time to do the process as Zolo is getting late for work.

Input:
The first line contains **N K**
The second line describes the current status of each bulb as an array whose each element can either be **G or Y or R**

Output:
The minimum amount of time required to clear the traffic jam.

Constraints:
1<=**N, K**<=100000

Example:
Input:
4 2
R Y G Y

Output:
5

****