**Sliding Maximum**

**You are given an array 'ARR' of integers of length 'N' and a positive integer 'K'. You need to find the maximum elements for each and every contiguous subarray of size K of the array.**

**For example**

**'ARR' = [3, 4, -1, 1, 5] and 'K' = 3**

**Output = [4, 4, 5]**

**Since the maximum element of the first subarray of length three ([3, 4, -1]) is 4, the maximum element of the second subarray of length three ([4, -1, 1]) is also 4 and the maximum element of the last subarray of length three [-1, 1, 5]) is 5, so you need to return [4, 4, 5].**

##### Input format:

**The first line of input contains a single integer 'T', representing the number of test cases or queries to be run. Then the 'T' test cases follow.**

**The first line of each test case contains two positive integers 'N' and 'K' which represent the length of the array and length of the subarray respectively.**

**The Second line of each test case contains 'N' space-separated integers representing the elements of the array.**

##### Output Format :

**For each test case, print 'X' space-separated integer denoting maximum elements for each and every contiguous subarray of size 'K' of the array. Where 'X' is the number of subarray of size 'K' in array 'arr'.**

**Output for each test case will be printed in a separate line.**

##### Constraint :

**1 <= T <= 10**

**1 <= N <= 10^5**

**-10^5 <= arr[i] <= 10^5**

**1 <= K <= N**

**Time Limit: 1 sec**

**You are given a string (STR) of length N.**

**Your task is to find the longest palindromic substring. If there is more than one palindromic substring with the maximum length, return the one with the smaller start index.**

**Note:**

**A substring is a contiguous segment of a string.**

**For example :**

**The longest palindromic substring of "ababc" is "aba", since "aba" is a palindrome and it is the longest substring of length 3 which is a palindrome. There is another palindromic substring of length 3 is "bab". Since starting index of "aba" is less than "bab", so "aba" is the answer.**

**Input Format:**

**The first line of input contains a single integer 'T', representing the number of test cases or queries to be run.**

**Then the 'T' test cases follow.**

**The first and only one of each test case contains a string (STR).**

**Output Format :**

**For every test case, print a single line containing the longest palindromic substring.**

**If there are multiple possible answers then you need to print the substring which has the lowest starting index.**

**Note :**

**You do not need to print anything; it has already been taken care of. Just implement the given function.**

**Follow up:**

**Try to solve it using O(1) space complexity.**

**Constraints :**

**1 <= T <= 10**

**0 <= N <= 10^3**

**where 'T' is the number of test cases, 'N' is the length of the given string.**

**Time Limit: 1 sec**

Jump Game

**You have been given an array 'ARR' of ‘N’ integers. You have to find the minimum number of jumps needed to reach the last index of the array i.e ‘N - 1’ if at any index ‘i’ we can jump to an index ‘i + k’ such that 1<= ‘k’ <= ARR[i] i.e the element you are currently at represents the maximum distance you can jump from the current element.**

**Your goal is to reach the last index in the minimum number of jumps.**

**Assume that, under the given constraints, you can always reach the last index.**

**Note:**

Consider 0 based indexing.

**Input format:**

The first line of input contains an integer ‘T’ denoting the number of test cases.

The first line of each test case contains an integer ‘N’ denoting the number of elements in the given sequence.

The second line of each test case contains ‘N’ space-separated integers denoting the elements in the sequence.

**Output Format:**

for each test case, print a single line containing a single integer denoting the minimum number of jumps required to reach the last index.

The output of each test case will be printed in a separate line.

##### Note:

You do not need to print anything, it has already been taken care of. Just implement the given function.

##### Constraints:

1 <= T <= 50

1 <= N <= 10 ^ 4

1 <= ARR[i] <= 10 ^ 4

Where ‘ARR[i]’ denotes the ‘i-th’ element of the ‘ARR’.

Time limit: 1 sec.

Design Question

They give a predefined file system project in which we have to write functionalities like to create/move/delete/copy a file. I have to use predefined classes to write my functionalities and build the file structure. They also ask to maintain multiple

Refer the below link:

<https://www.scribd.com/doc/100727494/CommVault-Questions>

1. There was a 2-D array of 'o' and 'x'. We have to rotate the array 90 degree and then bring all the 'x' in the bottom part of the 2-D matrix.

Input:  
o x o  
x x o  
o o x

Output:  
o o o  
o x o  
x x x

Explanation:  
Step 1- initial matrix  
o x o  
x x o  
o o x  
Step 2- Rotate it clockwise 90 degree  
o x o  
o x x  
x o o  
step 3- bring all the 'x' in the bottom part of the array  
o o o  
o x o  
x x x

1. Given two strings a and b, return the size of substring matching in both the strings

Input:  
String 1- ROMANINROMES  
String 2- XMANBATANINROMY

Output- 7

Explaination:

ROM"ANINROM"ES  
XMANBAT"ANINROM"Y  
the string in quotes are matching and their size is 7.

1. Find the number of turns to convert the string so that no two 'x' and 'y' are together.\*\*\*\*

Input:  
xxxyxyxyy

Output- 3

Explanation:  
xxxyxyxyyy -> xyxyxyxyxy

Maximum Path Sum in the matrix

**You have been given an N\*M matrix filled with integer numbers, find the maximum sum that can be obtained from a path starting from any cell in the first row to any cell in the last row.**

**From a cell in a row, you can move to another cell directly below that row, or diagonally below left or right. So from a particular cell (row, col), we can move in three directions i.e.**

Down: (row+1,col)

Down left diagonal: (row+1,col-1)

Down right diagonal: (row+1, col+1)

**Input format :**

The first line contains an integer 'T', which denotes the number of test cases or queries to be run. Then the test cases follow.

The first line of each test case contains two Integers 'N' and 'M' where 'N' denotes the number of rows in the given matrix. And 'M' denotes the number of columns in the given matrix.

The next 'N' line of each test case contains 'M' space-separated integers denoting the cell elements.

**Output format :**

For each test case/query, print the maximum sum that can be obtained by taking a path as described above.

Output for every test case will be printed in a separate line.

**Note :**

You do not need to print anything. It has already been taken care of.

**Constraints :**

1 <= T <= 50

1 <= N <= 100

1 <= M <= 100

-10^4 <= matrix[i][j] <= 10^4

Where 'T' is the number of test cases.

Where 'N' is the number of rows in the given matrix, and 'M' is the number of columns in the given matrix.

And, matrix[i][j] denotes the value at (i,j) cell in the matrix.

Time Limit: 1sec

Problem approach

The idea is to find maximum sum or all paths starting with every cell of first row and finally return maximum of all values in first row. We use Dynamic Programming as results of many subproblems are needed again and again.  
  
Time Complexity: O(n2).   
Auxiliary Space: O(n2).

#### Inplace rotate matrix 90 degree

You are given a square matrix of non-negative integers of size 'N x N'. Your task is to rotate that array by 90 degrees in an anti-clockwise direction without using any extra space.

**For example:**

For given 2D array :

[ [ 1, 2, 3 ],

[ 4, 5, 6 ],

[ 7, 8, 9 ] ]

After 90 degree rotation in anti clockwise direction, it will become:

[ [ 3, 6, 9 ],

[ 2, 5, 8 ],

[ 1, 4, 7 ] ]

**Input Format :**

The first line of input contains an integer 'T' representing the number of the test case. Then the test case follows.

The first line of each test case contains an integer 'N' representing the size of the square matrix ARR.

Each of the next 'N' lines contains 'N' space-separated integers representing the elements of the matrix 'ARR'.

**Output Format:**

For each test case, return the rotated matrix.

**Note:**

You do not need to print anything; it has already been taken care of. Just implement the given function.

**Constraints:**

1 ≤ T ≤ 50

1 ≤ N ≤ 100

1 ≤ ARR[i][j] ≤ 10^9

Time Limit: 1 sec

Rotate array

**Given an array *'arr'* with *'N'* elements, the task is to rotate the array to the left by *'K'* steps, where 'K' is non-negative.**

**Example:**

arr = [1,2,3,4,5]

k=1 rotated array = [2,3,4,5,1]

k=2 rotated array = [3,4,5,1,2]

k=3 rotated array = [4,5,1,2,3] and so on.

**Input Format:**

The first line contains an integer 'N' representing the size of the array.

The second line contains 'N' space-separated integers representing the elements of the array.

The last line contains an integer 'K' representing the number of times the array has to be rotated in the left direction.

**Output Format:**

The output contains 'N' space-separated integers representing the Rotated array elements.

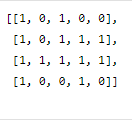
**Note:-**

You don’t need to print anything. Just implement the given function.

Maximum Area Square

**You have been given a non-empty grid ‘MAT’ consisting of only 0s and 1s. Your task is to find the area of maximum size square sub-matrix with all 1s.**

**If there is no such sub-matrix, print 0.**



**For example, for the following grid:**

The area of the largest square submatrix with all 1s is 4.

**Input Format:**

The first line contains an integer ‘T’ denoting the number of test cases. Then each test case follows.

The first input line of each test case contains two space-integers ‘N’ and ‘M’ representing the number of rows and columns of the grid, respectively.

From the second line of each test case, the next N lines represent the rows of the grid. Every row contains M single space-separated integers.

**Output Format:**

For each test case, print the area of maximum size square sub-matrix with all 1s.

Print the output of each test case in a separate line.

**Note:**

You are not required to print the expected output; it has already been taken care of. Just implement the function.

**Constraints:**

1 <= T <= 100

1 <= N <= 50

1 <= M <= 50

0 <= MAT[i][j] <= 1

Time limit: 1 sec

Count with K different characters

**You are given a string *'str'* of lowercase alphabets and an integer *'k'* .**

**Your task is to return the count all the possible substrings that have exactly 'k' distinct characters.**

**For example:**

'str' = abcad and 'k' = 2.

We can see that the substrings {ab, bc, ca, ad} are the only substrings with 2 distinct characters.

Therefore, the answer will be 4.

**Input format :**

The first line contains a string ‘str’.

The second line contains a single integer 'k'.

**Output format :**

The output contains a single integer which is the count of all possible substrings having exactly 'k' distinct characters.

**Note:**

You don’t have to print anything. Just implement the given function.

Collect Maximum Coins in Matrix

**You are given a matrix of ‘M’ rows and ‘N’ columns. The cells of the matrix contain either a coin or are empty.**

**You are allowed to visit every boundary cell that has a coin in it and collects that coin apart from that, you are allowed to collect the coin in one of the four adjacent cells. Find the maximum number of coins that you can collect from the matrix.**

**For Example :**

If Matrix of size 3 \* 5 is:

0 1 1 0 0

0 1 0 1 0

1 0 0 0 0

Then, out of the five coins in the matrix, you can collect a maximum of four coins. This is because the coin at (0, 1) lies on the boundary and after collecting the coin one can also collect the coin at (1, 1) as it lies in the adjacent cell. We can also collect the coin at (2, 0). But we cannot collect the coin at (1, 3), as this coin doesn’t lie on the boundary and it cannot be reached from one of the boundary coins.

**Input Format :**

The first line contains a single integer ‘T’ denoting the number of test cases, then each test case follows:

The first line of each test case contains two integers ‘M’ and ‘N’ denoting the number of rows and columns of the matrix.

The next M lines each contain N integers denoting the number of coins corresponding to each cell of the current row.

**Output Format :**

For each test case, print the maximum number of coins that we can collect.

Output for each test case will be printed in a separate line.

**Note :**

You are not required to print anything; it has already been taken care of. Just implement the function.

**Constraints :**

1 ≤ T ≤ 10

1 ≤ M, N ≤ 200

Matrix[i][j] = {0, 1}

Time limit: 1 sec

Excel Column Number

**You have been given a column title as appears in an Excel sheet, return its corresponding column number.**

**For example:**

A -> 1

B -> 2

C -> 3

...

Z -> 26

AA -> 27

AB -> 28

...

**Input Format**

The only line of input contains a string S i.e. column title

**Constraints:**

1 <= |S| <= 10

Input contains only uppercase English Alphabet letters

Time Limit : 1 sec

**Output Format**

The only line of output will print the column number corresponding to given column title

<https://www.codingninjas.com/studio/interview-experiences/commvault>

**Water Supply In A Village**

There are n houses in a village. You want to supply water for all the houses by building wells and by laying pipes.

For each house i we can either build a well inside it directly with cost of wells[i] or pipe in water from another well to it. The total cost to lay pipes between houses is given by pipes[i] = {house1, house2, cost} here pipes[i] represent the cost to lay a pipe between house1 and house2

You need to print the minimum total cost to supply water to all houses in the village.

**Input Format:**

The first line will contain an input n and k which represent the number of houses and the number of pipes respectively.

The following line will contain n space separated integers which will be the input for array wells

The following k line will contain input for array pipes

**Output Format:**

You need to print the minimum total cost to supply water to all houses in the village.

**Input:**

4 2

1 4 4 4

1 4 2

1 2 1

**Output:**

8

**Constraints:**

1 <= n <= 10 ^ 2

0 <= wells[i] <= 10^6

1 <= k <= 10000

1 <= pipes[i][0], pipes[i][1] <= N

0 <= pipes[i][2] <= 10^5

pipes[i][0] != pipes[i][1]

Middle Of Linked List

**Given a singly linked list of *'N'* nodes. The objective is to determine the middle node of a singly linked list. However, if the list has an even number of nodes, we return the second middle node.**

**Input Format :**

The first line contains an integer 'N', the size of the linked list.

The second line contains 'N' space-separated integers.

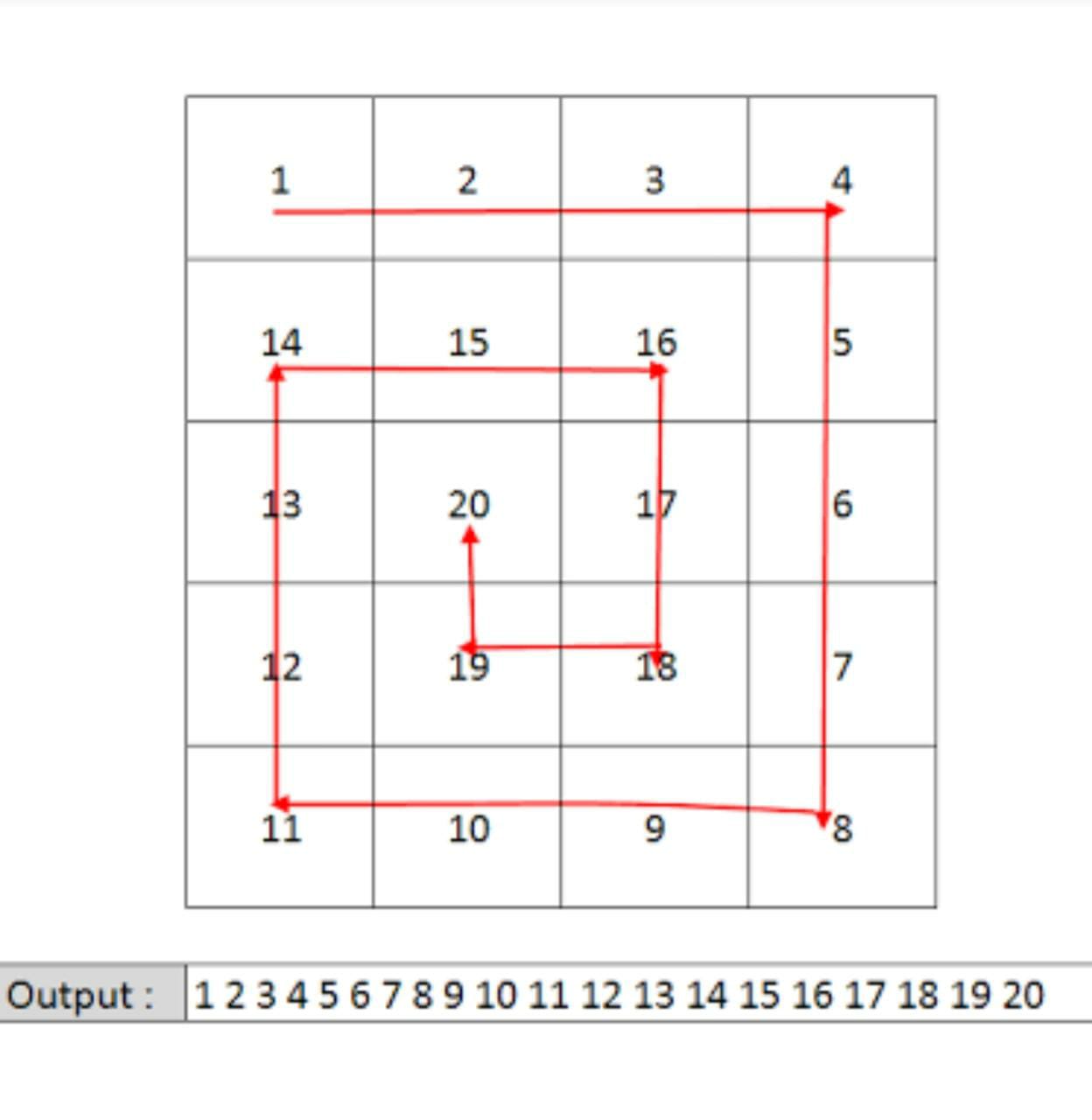
**Output Format :**

The output contains all the integers from the middle node.

Spiral Matrix

**You are given a 2-D array 'MATRIX' of dimensions N x M, of integers. You need to return the spiral path of the matrix.**

**Example Of Spiral Path:**



**Input Format:**

The first line contains an integer 'T' which denotes the number of test cases or queries to be run. Then the test cases follow.

The first line of each test case contains two single space-separated integers 'N' and 'M', denoting the number of rows and columns respectively.

The next 'N' lines, each contains 'M' single space-separated integers representing the elements in a row of the matrix.

**Output format :**

For each test case/query, print the spiral path of the given matrix.

The output for every test case will be printed in a separate line.

**Note:**

You do not need to print anything, it has already been taken care of. Just implement the given function.

**Constraints:**

1 <= T <= 5

1 <= N <= 10 ^ 2

1 <= M <= 10 ^ 2

-10 ^ 9 <= MATRIX[ i ][ j ] <= 10 ^ 9

Time Limit: 1sec.

Missing Number

**You are given an array/list ‘BINARYNUMS’ that consists of ‘N’ distinct strings which represent all integers from 0 to N in binary representation except one integer. This integer between 0 to ‘N’ whose binary representation is not present in list ‘BINARYNUMS’ is called ‘Missing Integer’.**

**Your task is to find the binary representation of that ‘Missing Integer’. You should return a string that represents this ‘Missing Integer’ in binary without leading zeros.**

**Note**

1. There will be no leading zeros in any string in the list ‘BINARYNUMS’.

**Example:**

Consider N = 5 and the list ‘binaryNums’= [“0”, “01”, “010”, “100”, “101”]. This list consists of the binary representation of numbers [0, 1, 2, 4, 5]. Clearly, the missing number is 3 and its binary representation will be “11”. So you should return string “11”.

**Input format:**

The first line of input contains an integer ‘T’ denoting the number of test cases. then ‘T’ test cases follow.

The first line contains single integers ‘N’ represent the size of the list ‘BINARYNUMS’.

The second line contains ‘N’ space-separated string representing the list ‘BINARYNUMS’.

**Output format :**

For each test case, print a single line containing a single string that represents this ‘Missing Integer’ in binary without leading zeros.

The output of each test case will be printed in a separate line.

**Note:**

You do not need to print anything, it has already been taken care of. Just implement the given function.

**Constraints:**

1 <= T <= 50

1 <= N <= 10 ^ 4

Where ‘T’ is the total number of test cases and ‘N’ is the size of list ‘BINARYNUMS’

Time limit: 1 sec.

Print Nodes at Distance K From a Given Node

**You are given an arbitrary binary tree, a node of the tree, and an integer 'K'. You need to find all such nodes which have a distance K from the given node and return the list of these nodes.**

**Distance between two nodes in a binary tree is defined as the number of connections/edges in the path between the two nodes.**

**Note:**

1. A binary tree is a tree in which each node has at most two children.

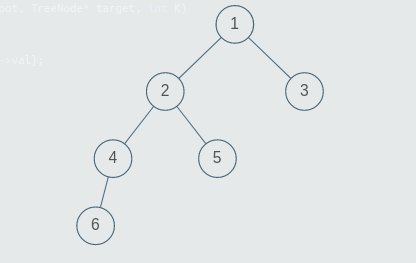
2. The given tree will be non-empty.

3. The given tree can have multiple nodes with the same value.

4. If there are no nodes in the tree which are at distance = K from the given node, return an empty list.

5. You can return the list of values of valid nodes in any order. For example if the valid nodes have values 1,2,3, then you can return {1,2,3} or {3,1,2} etc.

**Example :**



Consider this tree above. The target node is 5 and K = 3. The nodes at distance 1 from node 5 are {2}, nodes at distance 2 from node 5 are {1, 4} and nodes at distance 3 from node 5 are {6, 3}.

**Input Format:**

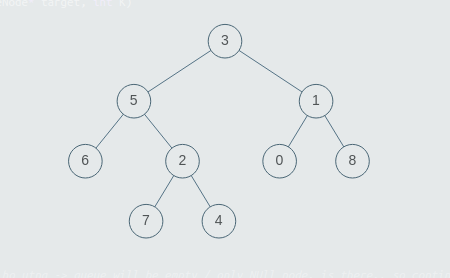
The first line will contain the values of the nodes of the tree in the level order form ( -1 for NULL node). Refer to the example below for further explanation.

The second line contains the value of the target node.

The third and the last line contains the integer K denoting the distance at which nodes are to be found.

Example:

Consider the binary tree:



The input for the tree depicted in the above image would be :

3

5 1

6 2 0 8

-1 -1 7 4 -1 -1 -1 -1

-1 -1 -1 -1

Explanation :

Level 1 :

The root node of the tree is 3

Level 2 :

Left child of 3 = 5

Right child of 3 = 1

Level 3 :

Left child of 5 = 6

Right child of 5 = 2

Left child of 1 = 0

Right child of 1 = 8

Level 4 :

Left child of 6 = null (-1)

Right child of 6 = null(-1)

Left child of 2 = 7

Right child of 2 = 4

Left child of 0 = null (-1)

Right child of 0 = null (-1)

Left child of 8 = null (-1)

Right child of 8 = null (-1)

Level 5 :

Left child of 7 = null (-1)

Right child of 7 = null (-1)

Left child of 4 = null (-1)

Right child of 4 = null (-1)

The first not-null node (of the previous level) is treated as the parent of the first two nodes of the current level. The second not-null node (of the previous level) is treated as the parent node for the next two nodes of the current level and so on.

The input ends when all nodes at the last level are null (-1).

**Note :**

The above format was just to provide clarity on how the input is formed for a given tree.

The sequence will be put together in a single line separated by a single space. Hence, for the above-depicted tree, the input will be given as:

3 5 1 6 2 0 8 -1 -1 7 4 -1 -1 -1 -1 -1 -1 -1 -1

**Output Format :**

Print the values of all nodes at distance = K, from the given target node.

**Note:**

You do not need to print anything, it has already been taken care of. Just implement the given function.

#### Minimum insertions to make a string palindrome

**A palindrome string is one that reads the same backward as well as forward.**

**You are given a string *'str'*.**

**Find the minimum number of characters needed to insert in 'str' to make it a palindromic string.**

**Example :**

Input: 'str' = "abca"

Output: 1

Explanation:

If we insert the character ‘b’ after ‘c’, we get the string "abcba", which is a palindromic string. Please note that there are also other ways possible.

**Input format:**

The first line contains a string 'str', containing lowercase English letters i.e from ‘a’ to ‘z’.

**Output format:**

The output contains a single integer denoting the minimum number of insertions needed to make 'str' palindrome.

**Note:**

You do not need to print anything; it has already been taken care of. Just implement the given function.

Pair Sum

**You are given an integer array 'ARR' of size 'N' and an integer 'S'. Your task is to return the list of all pairs of elements such that each sum of elements of each pair equals 'S'.**

**Note:**

Each pair should be sorted i.e the first value should be less than or equals to the second value.

Return the list of pairs sorted in non-decreasing order of their first value. In case if two pairs have the same first value, the pair with a smaller second value should come first.

**Input Format:**

The first line of input contains two space-separated integers 'N' and 'S', denoting the size of the input array and the value of 'S'.

The second and last line of input contains 'N' space-separated integers, denoting the elements of the input array: ARR[i] where 0 <= i < 'N'.

**Output Format:**

Print 'C' lines, each line contains one pair i.e two space-separated integers, where 'C' denotes the count of pairs having sum equals to given value 'S'.

**Note:**

You are not required to print the output, it has already been taken care of. Just implement the function.

**Constraints:**

1 <= N <= 10^4

-10^5 <= ARR[i] <= 10^5

-2 \* 10^5 <= S <= 2 \* 10^5

Time Limit: 1 sec

Special Sum

**You are given an array ‘ARR’ of length ‘N’. There are two operations, ‘FIRST\_SUM’ and ‘LAST\_SUM’ for every index ‘i’ (1 <= i <= N) in the array,**

**i) FIRST\_SUM(i) calculates the sum of first i numbers.**

**ii) LAST\_SUM(i) calculates the sum of last N-i+1 numbers.**

**Also for every ‘i’, SPECIAL\_SUM(i) can be calculated as:**

**SPECIAL\_SUM(i) = FIRST\_SUM(i) + LAST\_SUM(i)**

**Your task is to return the minimum SPECIAL\_SUM for 0 <= i <= N - 1.**

**For example:**

Given ‘N’ = 4 and ‘ARR’ = [1, 2, 3, 4].

Then the minimum special sum will be 5 for i = 0 (0-based indexing), which is (1 + 4) = 5.Sum of 1 integer from beginning and end.

For i = 1 it will be (1 + 2) + (3 + 4) = 10

For i = 2 it will be (1 + 2 + 3) + (2 + 3 + 4) = 15

For i = 3 it will be (1 + 2 + 3 + 4) + (1 + 2 + 3 + 4) = 20

All of which are greater than 5.

**Input format:**

The first line of input contains an integer ‘T’ denoting the number of test cases.

The first line of each test case contains a single integer N, where ‘N’ is the number of elements of the array.

The second line of each test case contains ‘N’ space-separated integers, denoting the array elements.

**Output format:**

For each test case, return the minimum SPECIAL\_SUM for ‘i’ in the range [ 0, N-1 ].

The output of each test case will be printed in a separate line.

**Note:**

You don’t need to print anything. You just need to implement the given function.

**Constraints:**

1 <= T <= 5

1 <= N <= 5 \*10^3

-5 \*10^2 <= ARR[i] < 5 \*10^2

Time limit: 1 sec

URL Shortener

**You have given a URL id – 'N' consisting of only digits. Your task is to generate a short URL for the given URL id.**

**To generate a short URL, you need to convert the given URL id to 62 base number where digits of the number are:[“0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ”] i.e “0” corresponds to 0, “a” corresponds to 10, “Z” corresponds to 61, and “10” corresponds to 62, “11” corresponds to 63, and so on….**

**Follow Up:**

Can you solve this in logarithmic time and space complexity?

**Input format :**

The first line of input contains a single integer 'T', representing the number of test cases.

The first line of each test contains a single integer 'N', denoting the URL id.

**Output format :**

For each test case, return a short URL for the URL id.

**Note:**

You don't need to print anything, it has already been taken care of. Just implement the given function.

**Constraints :**

1 <= T <= 1000

0 <= N <= 10^18

Time limit: 1sec

Check If The String Is A Palindrome

**You are given a string 'S'. Your task is to check whether the string is palindrome or not. For checking palindrome, consider alphabets and numbers only and ignore the symbols and whitespaces.**

**Note :**

String 'S' is NOT case sensitive.

**Example :**

Let S = “c1 O$d@eeD o1c”.

If we ignore the special characters, whitespaces and convert all uppercase letters to lowercase, we get S = “c1odeedo1c”, which is a palindrome. Hence, the given string is also a palindrome.

**Input format :**

The very first line of input contains an integer 'T' denoting the number of test cases.

The first line of every test case contains the string 'S'.

**Output format :**

For each test case, print “Yes” if 'S' is a palindrome, and “No” otherwise.

Print the output of each test case in a separate line.

**Note :**

You do not need to print anything, it has already been taken care of. Just implement the given function.

**Follow Up :**

Can you solve the problem using O(1) space complexity?

**Constraints :**

1 <= T <= 100

1 <= Length(S) <= 10^4

Where 'T' denotes the number of test cases and 'S' denotes the given string.

Time Limit : 1 sec

Rat In A Maze

##### You are given a starting position for a rat which is stuck in a maze at an initial point (0, 0) (the maze can be thought of as a 2-dimensional plane). The maze would be given in the form of a square matrix of order 'N' \* 'N' where the cells with value 0 represent the maze’s blocked locations while value 1 is the open/available path that the rat can take to reach its destination. The rat's destination is at ('N' - 1, 'N' - 1). Your task is to find all the possible paths that the rat can take to reach from source to destination in the maze. The possible directions that it can take to move in the maze are 'U'(up) i.e. (x, y - 1) , 'D'(down) i.e. (x, y + 1) , 'L' (left) i.e. (x - 1, y), 'R' (right) i.e. (x + 1, y).

**Note:**

Here, sorted paths mean that the expected output should be in alphabetical order.

**For Example:**

Given a square matrix of size 4\*4 (i.e. here 'N' = 4):

1 0 0 0

1 1 0 0

1 1 0 0

0 1 1 1

Expected Output:

DDRDRR DRDDRR

i.e. Path-1: DDRDRR and Path-2: DRDDRR

The rat can reach the destination at (3, 3) from (0, 0) by two paths, i.e. DRDDRR and DDRDRR when printed in sorted order, we get DDRDRR DRDDRR.

**Input format:**

The first line contains an integer 'N', which denotes the dimensions of the square matrix (maze).

Then 'N' lines follow. Each line contains 'N' space-separated integers denoting the values which would either be 0 denoting a blocked path or 1 denoting the available path in the maze, respectively.

**Output format:**

For the given maze, print the vector/list of strings representing all the possible paths that the rat can take to reach from source to destination in the maze in sorted order.

Output for each test case will be printed in a separate line.

**Note:**

You do not need to print anything. It has already been taken care of. Just implement the given function.

**Constraints:**

2 <= N <= 5

0 <= MATRIX[i][j] <= 1

Where N is the size of the square matrix.

Time Limit: 1sec

<https://www.codingninjas.com/studio/interview-experiences/darwinbox>

Data Structures, Algorithms, System Design, Aptitude, OOPS

Array, Linked List, Tree, Graph, Dynamic Programming, Stacks, Recursion