

QUESTION BANK DOCKET

**Table of Contents**

[1 Source 2](#_Toc253285289)

[2 Category 3](#_Toc965934774)

[3 Tech Area 3](#_Toc613746458)

[4 Complexity 3](#_Toc904143571)

[5 Question 3](#_Toc1426514880)

[6 Answer 3](#_Toc441207452)

[7 References 3](#_Toc1838997797)

# Source

Enter the name of the source where you got the question from

CodeByte

# Category

Programming, **Java,** .NET, SQL, Angular

(Mark one of the above as Bold and Underline)

# Tech Area

As given in the excel for respective categories

Data Structure

# Complexity

Simple / Medium / Complex

# Question

Paste here

**Ghanavi Question**

**1.Problem**

**Find Intersection**

Have the function Find Intersection(strArr) read the array of strings stored in strArr which will contain 2 elements: the first element will represent a list of comma-separated numbers sorted in ascending order, the second element will represent a second list of comma-separated numbers (also sorted). Your goal is to return a comma-separated string containing the numbers that occur in elements of strArr in sorted order. If there is no intersection, return the string false.

**2.Problem**

The challenge is to find all the pairs of two integers in an unsorted array that sum up to a given S.

For example, if the array is [3, 5, 2, -4, 8, 11] and the sum is 7, your program should return [[11, -4], [2, 5]] because 11 + -4 = 7 and 2 + 5 = 7.

**3.Problem**

3)Longest Word

Have the function LongestWord(sen) take the sen parameter being passed and return the longest word in the string. If there are two or more words that are the same length, return the first word from the string with that length. Ignore punctuation and assume sen will not be empty. Words may also contain numbers, for example "Hello world123 567"

Examples

Input: "fun&!! time"

Output: time

Input: "I love dogs"

Output: love

**4.Problem**

Questions Marks

Have the function QuestionsMarks(str) take the str string parameter, which will contain single digit numbers, letters, and question marks, and check if there are exactly 3 question marks between every pair of two numbers that add up to 10. If so, then your program should return the string true, otherwise it should return the string false. If there aren't any two numbers that add up to 10 in the string, then your program should return false as well.

For example: if str is "arrb6???4xxbl5???eee5" then your program should return true because there are exactly 3 question marks between 6 and 4, and 3 question marks between 5 and 5 at the end of the string.

Examples

Input: "aa6?9"

Output: false

Input: "acc?7??sss?3rr1??????5"

Output: true

**5.Problem**

Night king wants to attack the kingdom and he really likes cold days. However, he doesn’t want to attack the kingdom on the coldest day, because it is obvious.

Instead, he will attack the second coldest day. Given an array A*A* of N*N* **distinct** integers where *Ai*​ represents the temperature forecast of the *i*-th day, You need to find the temperature of the day of the attack.

**Input Format**

* First-line will contain T, the number of test cases. Then the test cases follow.
* Each test case contains two lines of input.
* The first line of every test case contains an integer *N* - the number of days
* The second line of every test case contains N integers *A*1​,*A*2​,..,*AN*​ denoting the temperature forecast of the *i*-th day.

**Output Format**

For each test case, output in a single line - the answer to the *i*-th test case.

**Constraints**

* 1≤*T*≤1500
* 2≤*N*≤2⋅10^5
* 1≤*Ai*​≤10^9

**Sample 1:**

Input

3

2

1 2

3

7 4 9

5

45 76 91 21 9

Output

2

7

21

**6.Problem**

You are given T testcases , in each testcase you are given three numbers A*A*, B*B* and C*C* .  
Find that whether an expression of the form ‘‘*xAyBzC*" exists ( where x , y and z can be + or - ) , such that the final result is 00. If it exists print ‘‘*YES*" (without quotes), else print ``NO"(without quotes).

* **Input Format**
* First line contains T , Number of Testcases
* Now each test case contains 3 integers A,B,C
* **Output Format**
* For each testcase print one line consisting of ‘‘*YES*" or ‘‘*NO*" (without quotes) as described in problem.  
  Output is Case-Insensitive i.e. ‘‘*Yes*" , ‘‘*nO*" , ‘‘*YeS*" all will be considered correct.
* **Constraints**
* 1≤*T*≤100000
* 1≤*A*,*B*,*C*≤10^18

**Sample 1:**

Input

2

1 1 1

1000000000000000000 1000000000000000000 1000000000000000

Output

NO

NO

**Explanation:**

We can't make any expression using + or - in both the cases such that the final result equals 0.

**7. Problem**

You are given T testcases , in each testcase you are given a binary string S*S* of length N*N* consisting of only 0's and 7's . Find the number of subsequence that are equal to ``007".  
A subsequence of a string is a string that is generated by deleting some character of the given string without changing its order.

**Input Format**

* First line contains T , Number of Testcases
* Next T lines contains a binary string S of length N of each Testcase.

**Output Format**

* For each testcase print one line consisting of number of subsequence’s ``007"

**Constraints**

* 1≤*T*≤100
* 1≤*N*≤10^5
* Sum of N*N* overall Testcases does not exceed 10^6106

**8. Problem**

You are given a sequence *A*1​,*A*2​,…,*AN*​.

A **peak** is a subarray (a contiguous subsequence) of A*A* whose elements first strictly increase and then strictly decrease in value.

More formally, a subarray from L to R is a **peak** if and only if there exists an integer X such that *L*≤*X*≤*R* for which the subarray from L to X is strictly increasing and the subarray from X to R is strictly decreasing. Note that subarrays that are only strictly increasing and subarrays that are only strictly decreasing are considered **peaks** as well.

Find the maximum length of a **peak** in the sequence A.

**Input Format**

* The first line of the input contains a single integer Tdenoting the number of test cases. The description of T test cases follows.
* The first line of each test case contains an integer N.
* The second line contains N*N* space-separated integers *A*1​,*A*2​,…,*AN*​.

**Output Format**

* For each test case, print a single line containing one integer ― the maximum length of a **peak** in the sequence A*A*.

**Constraints**

* 1≤*T*≤200
* 1≤*N*≤2⋅10^5
* the sum of N*N* over all test cases does not exceed 4⋅10^5
* 1≤*Ai*​≤10^9 for each valid i

**Sample 1:**

Input

4

6

1 3 5 3 1 9

7

3 5 4 7 5 6 3

4

1 5 7 8

3

9 5 1

output

5

3

4

3

**Explanation:**

**Example case 1:** The longest peak is of length 55. One such subarray is from position 11 to position 55 in the sequence.

**Example case 2:** The longest peak is of length 33. One such subarray is from position 11 to position 33 in the sequence.

**Example case 3:** The longest peak is of length 44. One such subarray is from position 11 to position 44 in the sequence. Although it is strictly increasing, it is still considered a peak.

**Example case 4:** The longest peak is of length 33. One such subarray is from position 11 to position 33 in the sequence. Although it is strictly decreasing, it is still considered a peak.

**9. Problem**

You are given a positive integer X. You can perform the following two types of operations on it any number of times(possibly, zero):

* Choose a positive integer K, and then multiply X with K.
* If X is a perfect square, change Xto square root *X*​.

Find the minimum value of X*X* you can achieve.

**Input Format**

* First-line will contain T*T*, the number of test cases. Then the test cases follow.
* Each test case contains a single line of input, an integer X*X*.

**Output Format**

For each test case, output in a single line - the minimum possible value of X*X*

**Constraints**

* 1≤*T*≤2000
* 1≤*X*≤10^9

**Sample 1:**

Input

3

3

4

40

output

3

2

10

**10. Problem**

Alice has been asked to balance a string S*S* given to her. The string contains of only ‘(‘ and ‘)’. A string S*S* is said to be balanced if-

* Open brackets must be closed by the corresponding closing bracket.
* Open brackets must be closed in the correct order.

Find the minimum number of ‘(‘ or ‘)’ she would have to add anywhere to balance out the string.

**Input Format**

* First line will contain T, the number of test cases.
* For every test case, next line will contain 1 integer N ,size of the string And its next line has the string S.

**Output Format**

For each test case, the minimum number of parentheses Alice can add to balance the string S. separated by a new line or a space.

**Constraints**

* 1≤*T*≤10

1≤∣*S*∣≤10^6

**Dhanyashree**

**1. Problem (Simple)**

A monster has H health points. Each time you hit it with a sword it loses X health points. However, the monster always gains Y health points right before every one of your hits.

The monster is considered defeated when the number of health points it has becomes zero or less than zero. You need to find if it is possible to defeat the monster.

**Input Format**

The first line of the input contains a single integer T - the number of test cases. The description of T test cases follows.

The first line of each test case contains three space-separated integers H, X and Y.

**Output Format**

For each test case, print a single line containing one integer. That integer should be 1 if it is possible to defeat the monster and 0 otherwise.

Constraints

* 1≤*T*≤100
* 1≤*H,X,Y*≤10^18

**Sample 1:**

Input Output

4 1

3 6 2 1

4 6 3 0

7 1 2 0

1 1 2

**Explanation:**

**Example case 1**: The monster will have 3+2=53+2=5 health points before the first hit. Therefore, it will be defeated after the first hit.

Example case 2: The monster will have 4+3=74+3=7 health points before the first hit. After the first hit, the monster will have 7-6=17−6=1 health point. Before the second hit, the monster will have 1+3=41+3=4 health points. Therefore, it will be defeated after the second hit.

**Example case 3:** It is impossible to defeat the monster.

**Example case 4:** It is impossible to defeat the monster.

**2. Problem (Intermediate)**

During the assembly, a teacher lines up N*N* students in a line in front of her. She can see the first student and all the students taller than the students before them ( i.e a teacher can see the student i if *hi* >*hj* for all i*j*≤*i* ). How many students will the teacher be able to see?

**Input Format**

* The first line will have 1 integer N.
* The second line will contain the heights of the students which would be integers in the order of them getting farther from the teacher. (will be separated by a space)

**Output Format**

Output a single integer which is the amount of students the teacher will be able to see.

**Constraints**

**1≤*N*≤105**

**2≤*hi* ≤109**

### Sample 1:

Input Output

9 91 2 3 4 5 6 7 8 9

**Explanation:**

The teacher will be able to see all the students.

### Sample 2:

Input Output

4 31 2 4 3

**Explanation:**

The teacher will not be able to see the 4th student as the height of the 3rd student has height higher than him.

**3. Problem(Intermediate)**

You are given a sequence *A*1 ,*A*2 ,…,*AN* whose elements are zeros and ones.

A sequence that consists of *L* integers is called a permutation if and only if each integer from 1 to i appears exactly once in it.

You need to find any permutation P of integers from 1 to Nfor which it holds that for every valid index i, *Ai* =1 if the prefix of P of length Iis a permutation of numbers from 1 to Iand *Ai* =0 otherwise, or determine that such a permutation does not exist.

**Input Format**

* The first line of the input contains a single integer T*T* - the number of test cases. The description of T*T* test cases follows.
* The first line of each test case contains a single integer N*N*.
* The second line of each test case contains N*N* space-separated integers *A*1 ,*A*2 ,…,*AN* .

**Output Format**

* For each test case, print a single line.
* In case a suitable permutation P*P* does not exist, print a single integer −1.
* Otherwise, print N space-separated integers *P*1 ,*P*2 ,…,*PN* .

### Constraints

1 \leq T \leq 1001≤*T*≤100

1 \leq N \leq 2 \cdot 10^51≤*N*≤2⋅105

A\_i=0*Ai* =0 or A\_i=1*Ai* =1 for each valid i*i*

A\_N=1*AN* =1

the sum of N*N* over all test cases does not exceed 4 \cdot 10^54⋅105

### Sample 1:

Input Output

4 1 3 2 44 1 2 3 4 1 0 1 1 3 1 2 7 4 5 6 84 3 1 21 1 1 180 0 1 0 0 0 1 130 0 1

**Explanation:**

**Example case 1:** The prefixes of lengths 11, 33 and 44 are valid permutations and the prefix of length 22 is not a valid permutation.

**Example case 2:** This is the only valid permutation such that all its prefixes are valid permutations.

**Example case 3:** The prefixes of lengths 33, 77 and 88 are valid permutations and the rest of the prefixes are not valid permutations.

**Example case 4:** Only the prefix of length 33 is a valid permutation.

**4. Problem(Simple)**

You are given an integer X. Find the smallest number Ysuch that :

* *Y* is greater than X.
* All digits of *Y* are pairwise different.

**Input Format**

* First-line will contain T, the number of test cases. Then the test cases follow.
* Each test case contains a single line of input, a single integer *X*.

**Output Format**

For each test case, output in a single line the smallest integer *Y* which satisfies the given conditions.

* **Constraints**
* 1≤*T*≤2000
* 1≤*X*≤10^5

### Sample 1:

Input Output

3 65 109 1023990

**Explanation:**

* **Test Case 11:** 66 is the smallest integer which is greater than 55, and it also satisfies the other conditions it is a single-digit integer.
* **Test Case 33:** It can be proved that 10231023 is the smallest integer which satisfies the given conditions

**5.Problem(Array intermediate)**

Chef will give you an array A*A* of length N*N* which is a **permutation**. You can split the given array into some number of partitions, individually sort each subarray and join them back together without changing their relative order. After concatenating them, the result should a the sorted array. Find the ***maximum*** number of partitions we can make.

**Note:** A permutation is an array consisting of N*N* **distinct** integers from 11 to N*N* in **arbitrary** order.

**Input Format**

* First-line will contain T*T*, the number of test cases. Then the test cases follow.
* The first line of each test case contains an integer N*N* - the length of the array.
* The second line of each test case contains N*N* integers - A\_1,A\_2,..,A\_N*A*1 ,*A*2 ,..,*AN* denoting the integers of the array.

**Output Format**

For each testcase, output in a single line - the maximum number of partitions possible.

**Constraints**

* 1≤*T*≤1500
* 1≤*N*≤2⋅10^5,∑*N*≤5⋅10^5
* 1≤*Ai* ≤*N*
* Ais a permutation.

### Sample 1:

Input Output

3 46 33 2 1 4 5 6 131 2 385 8 7 3 6 1 2 4

**Explanation:**

* **Test Case 11**: We can partition the array as follows - [3,2,1] [4] [5] [6][3,2,1][4][5][6], so we will have 44 partitions.
* **Test Case 22**: We can partition the array as follows - [1] [2] [3][1][2][3], so we will have 33 partitions.

**6. Problem(Array Complex)**

Given an array A (11-indexed) of positive integers size N and Q queries. Each query will be in the form l *r* x which means multiply all the elements in array A from index l to r (inclusive) by x. Output the maximum power of 2 in the final array A.

**Input Format**

* The first line of input contains 2 integers N - size of array A and Q- number of queries
* The second line contains N*N* space-separated integers *A*1 ,*A*2 ,……,*AN*
* Each of the next Q lines contains three space-separated integers l r x denoting the query.

**Output Format**

Output a single line containing one integer - Maximum power of 2 in the final array.

**Constraints**

* 1≤*N*≤10^5
* 1≤*Q*≤10^5
* 1≤*l*≤*r*≤*N*
* 2≤*x*≤10^9
* 1≤*Ai* ≤10^9

### Sample 1:

Input Output

5 4 41 2 3 4 51 1 32 3 43 3 61 5 2

**Explanation:**

Query 1 : Multiply numbers from index 1 to 1 by 3 Resulting array : 3 2 3 4 5

Query 2 : Multiply numbers from index 2 to 3 by 4 Resulting array : 3 8 12 4 5

Query 3 : Multiply numbers from index 3 to 3 by 6 Resulting array : 3 8 72 4 5

Query 4 : Multiply numbers from index 1 to 5 by 2 Resulting array : 6 16 144 8 10

So, highest power of 22 in the array is 44 as 144144 has prime factorisation as 2^4⋅3^2.

**7. Problem(String)**

You and Arup were arguing for who is a better problem solver, to resolve the matter you two came to Aman and decided that his decision will be considered final. Aman gave you both a string S of length N, and asked you both to find the length of the longest substring X of the array such that the reverse(X) = X*reverse*(*X*)=*X*. Arup failed to solve the task, can you solve the task?

**Input Format**

* The first line contains an integer T, representing T test cases.
* First line of each test case contains an integer N , representing the size of string S.
* Second line of each test case contains the string S of size N.

**Output Format**

* For each test case, print a single integer representing the length of the longest substring X.

**Constraints**

* 1≤T≤100
* 1≤N≤1000
* S contains only lowercase English letters.
* Sum of N over all test cases does not exceed 10^4

### Sample 1:

Input Output

3 33Aba 14 abcd6 4abbacc

**Explanation:**

In the first test case, string aba*aba* of length 3 remains the same even after its reversed.  
 In the second test case, string b*b* of length 1 remains the same even after its reversed. Other strings of length 1 are also present which even after getting reversed remains the same.  
 In the first test case, string abba*abba* of length 4 remains the same even after its reversed.

**8. Problem**

You are given an integer N and a string S, whose length is N. Your task is to split the string S into k consecutive substrings of the same length, so that the splitting is magical, while maximizing the k.

We call a splitting magical if all of the substrings are anagrams of each other. We say that two string A and B are anagrams of each other if we can rearrange the characters of the string A so that it becomes equal to string B.

**Input Format**

* First line will contain T, number of testcases. Then the testcases follow.
* Each testcase contains of a single line of input, an integer N and a string S.

**Output Format**

For each testcase, output in a single line maximal possible value of k*k*. It can be showed that the answer always exists.

**Constraints**

* 1≤*T*≤100
* 1≤*N*≤10^5
* Sum of N*N* over all test cases does not exceed 2⋅10^5

### Sample 1:

Input Output

3 36 abbaab 44 aaaa 16 abcdef

**Explanation:**

In the first test case, it is optimal to split the string into ab, ba and ab since they are all anagrams of (for example) ab.

In the second test case, it is optimal to split the string into a, a, a, a.

In the third test case, the string cannot be split into more than one component, which is the string itself abcdef.

**9. Problem**

You are given a singly linked list. Complete two functions first for deleting the node with the given value and second for inserting the value.

Node\* delete(Node\* head, int position)

Node\* insert(Node\* head, int position, int value)

**Input Format**

* The first line of input contains 2 integers N - length of linked list and Q - number of queries
* Second line contains N*N* space separated integers *val*1 ,*val*2 ,....*vali* ,...*valn* where *vali* is the value stored at ith node starting from the head node.
* Each of the next Q lines contains either
  + 2 integers 0 and *pos*, where 0 denotes delete operation and pos is the position of node to be deleted.
  + or 3 integers 1 , *pos* and *value*, where 11 denotes insert operation, pos is the position where node should be inserted and value is the value of the inserted node.

**Note:**

* For C++ language, you need to:

Complete the function in the submit solution tab:

Node\* remove(Node\* head, int position){..}

Node\* insert(Node\* head, int position, int value){..}

For both the functions return pointer to the head of the linked list.

**Output Format**

Using the functions you have completed, the linked list generated after each query should be correct.

* For each query K*K* space separated integer val\_1, val\_2,.... val\_i,... val\_k*val*1 ,*val*2 ,....*vali* ,...*valk* after executing the query will be outputted. Here val\_i*vali* is the value of stored at ith node starting from head and K*K* is the length of linked list after executing the query.

**Constraints**

* 1≤*N*≤10^4
* 1≤*Q*≤100
* 1≤*val*≤10^5

**Sample 1:**

Input

3 5

10 20 5

0 2

1 3 12

1 2 11

1 1 13

0 5

Output

10 5

10 5 12

10 11 5 12

13 10 11 5 12

13 10 11 5

**Explanation:**

Initial Linked List : 10 → 20 → 5

Query 1 : 0 2 Delete 2*nd* node, so new linked list : 10 → 5

Query 2 : 1 3 12 Add node with value 12 at 3*rd* position, so new linked list : 10 → 5 → 12

Query 3 : 1 2 11 Add node with value 11 at 2*nd* position, so new linked list : 10 → 11 → 5 → 12

Query 4 : 1 1 13 Add node with value 13 at 1*st* position, so new linked list : 13 → 10 → 11 → 5 → 12

Query 5 : 0 5 Delete last node, so new linked list : 13 → 10 → 11 → 5

**10. Problem**

Given the linked list, If there is any loop in the linked list return the number of elements in the loop else return −1.

**Input Format**

* First-line contains one integer N, the number of elements in the linked list.
* Second-line contains a number M which means tail nodes point to the *Mth* node.1*M*=−1 if there is no loop.
* **Note:** All node values are 0 and 0-based indexing is used (i.e. head is 0*th* node).
* For C++ language, you need to:

Complete the function in the submit solution tab:

int solve(Node\* head){}

**Output Format**

If your code is correct then the final code will print −1 if a loop doesn't exist, else the length of the loop.

* **Constraints**

1≤*N*≤105

−1≤*M*<*N*

### Sample 1:

Input

62 Output4

**Mohan Pilantre**

1) Complete the code in your editor by writing an overridden getNumberOfTeamMembers method that prints the same statement as the superclass' getNumberOfTeamMembers method, except that it replaces n with 11 (the number of players on a Soccer team)

2) The Java *instanceof* operator is used to test if the object or instance is an instanceof the specified type.

In this problem we have given you three classes in the editor:

* Student class
* Rockstar class
* Hacker class

In the main method, we populated an *ArrayList* with several instances of these classes. *count* method calculates how many instances of each type is present in the ArrayList. The code prints three integers, the number of instance of Student class, the number of instance of Rockstar class, the number of instance of Hacker class.

3) Write the following methods that *return a lambda expression* performing a specified action:

1. PerformOperation isOdd(): The lambda expression must return True if a number is odd or False if it is even.
2. PerformOperation isPrime(): The lambda expression must return True if a number is prime or False if it is composite.
3. PerformOperation isPalindrome(): The lambda expression must return True if a number is a palindrome or False if it is not.

4) Accept a date (day, month, year) and display how many days in the year have gone by till the current date

//Ex:

//Day = 27 03 2022

//No of Days

//Jan 31, Feb 28, Mar 27 so total 🟰 86

//In case of a leap year Feb will have 29 days

5) Given an array, S , of N real number strings, sort them in descending order — but wait, there's more! Each number must be printed in the exact same format as it was read from stdin, meaning that .1 is printed as .1 , and 0.1 is printed as 0.1 . If two numbers represent numerically equivalent values (e.g., .1 ==0.1 ), then they must be listed in the same order as they were received as input).

Complete the code in the unlocked section of the editor below. You must rearrange array 's elements according to the instructions above.

**Input Format**

The first line consists of a single integer, N, denoting the number of integer strings.  
 Each line i of the N subsequent lines contains a real number denoting the value of s .

**Constraints**

* 1<=n<=200
* Each s has *at most 300*  digits.

6) Given a string of text in a tag-based language, parse this text and retrieve the contents enclosed within sequences of well-organized tags meeting the following criterion:

1. The name of the *start* and *end* tags must be same. The HTML code <h1>Hello World</h2> is *not valid*, because the text starts with an h1 tag and ends with a non-matching h2 tag.
2. Tags can be nested, but content between nested tags is considered *not valid*. For example, in <h1><a>contents</a>invalid</h1>, contents is *valid* but invalid is *not valid*.
3. Tags can consist of any printable characters.

**Input Format**

The first line of input contains a single integer, N (the number of lines).  
 The N subsequent lines each contain a line of text.

**Constraints**

* 1<=N<=100
* Each line contains a maximum of 104 printable characters.
* The total number of characters in all test cases will not exceed 106.

**Output Format**

For each line, print the content enclosed within valid tags.  
 If a line contains multiple instances of valid content, print out each instance of valid content on a new line; if no valid content is found, print None.

7) Write a class called *MyRegex* which will contain a string pattern. You need to write a regular expression and assign it to the pattern such that it can be used to validate an IP address. Use the following definition of an IP address:

IP address is a string in the form "A.B.C.D", where the value of A, B, C, and D may range from 0 to 255. Leading zeros are allowed. The length of A, B, C, or D can't be greater than 3.

Some valid IP address:

000.12.12.034

121.234.12.12

23.45.12.56

Some invalid IP address:

000.12.234.23.23

666.666.23.23

.213.123.23.32

23.45.22.32.

I.Am.not.an. ip

In this problem you will be provided strings containing any combination of ASCII characters. You have to write a regular expression to find the valid IPs.

Just write the MyRegex class which contains a String Patten. The string should contain the correct regular expression.

(MyRegex class *MUST NOT* be public)

8) You have an array ***A*** of ***N*** integers ***A1 A2 .. An***. Find the longest increasing subsequence ***Ai1 Ai2 .. Ak***  
(1 <= k <= N) that satisfies the following condition:  
 For every adjacent pair of numbers of the chosen subsequence ***Ai[x]*** and ***Ai[x+1]*** (1 < x < k***),*** the expression( ***Ai[x]*** & ***Ai[x+1]*** ) \* 2 < ( ***Ai[x]*** | ***Ai[x+1]*** ) is true

**Note**: ‘&’ is the bitwise AND operation, ‘ | ‘ is the bit-wise OR operation

**Input:**

The first line contains an integer, N, denoting the number of elements in A.

Each line i of the N subsequent lines (where 0 ≤ i < N) contains an integer describing Ai.

9) You have been given a string S of length N. The given string is a binary string which consists of only 0’s and ‘1’s. Ugliness of a string is defined as the decimal number that this binary string represents.

**Example:**

* “101” represents 5.
* “0000” represents 0.
* “01010” represents 10.

There are two types of operations that can be performed on the given string.

* Swap any two characters by paying a cost of A coins.
* Flip any character by paying a cost of B coins
* flipping a character means converting a ‘1’to a ‘0’or converting a ‘0’ to a ‘1’.

10) You are given an array a of N Integers and asked to split the array a into k consecutive segments such that each element of a belongs to exactly one segment and the sum of the cost of all segments is minimum.

We define the cost of some segment t as the sum of distances between the first and last occurrence for each unique element in the segment t.

Your task is to find the minimum possible total sum of the cost of all segments.

**Input Format**

* The first line contains an integer, n, denoting the number of elements in a.
* The next line contains an integer, k, denoting the Number of required consecutive segments..
* Each line i of the n subsequent lines (where 0 <=i<n) contains an integer describing a[i].

**Constraints :**

* 1 <= n<= 35000
* 1 <=k <=min(n,100)
* 1 <=a[i] <=n

**Swadhish**

1. **Stickler Thief Problem.**

Stickler the thief wants to loot money from a societyhaving **n** houses in a single line. He is a weird person and follows a certain **rule** when looting the houses. According to the rule, he will **never loot two consecutive houses**. At the same time, he wants to **maximize** the amount he **loots**. The thief knows which house has what amount of money but is unable to come up with an optimal looting strategy. He asks for your help to **find the maximum money he can get** if he strictly **follows** the **rule**. Each house has **a[i]amount of money** present in it.

**Example 1:**

**Input:**

n = 6

a[] = {5,5,10,100,10,5}

**Output:** 110

**Explanation:** 5+100+5=110

**Example 2:**

**Input:**

n = 3

a[] = {1,2,3}

**Output:** 4

**Explanation:** 1+3=4

**Your Task:**  
 **Complete the functionFindMaxSum()**which takes an array **arr[]** and **n** as input which returns the maximum money he can get following the rules

**Expected Time Complexity:**O(N).  
 **Expected Space Complexity:**O(N).

**Constraints:**  
 **1 ≤ n ≤ 10**4  
 1 ≤ a[i] ≤ 104

**2**. Java provides an inbuilt object type called **Stack**. It is a collection that is based on the last in first out (LIFO) principle. Try this problem using Stack.

Given **n** elements of a stack **st** where the first value is the bottom-most value of the stack and the last one is the element at top of the stack, delete the middle element of the stack without using any additional data structure.

**Example 1:**

**Input:** n = 5st = {1, 2, 3, 4, 5}**Output:** 5 4 2 1**Explaination:** The middle element is 3. If it is deleted and then the values are seen from top, this will be the order.

**Example 2:**

**Input:** n = 6st = {1, 4, 9, 2, 6, 5}**Output:** 5 6 2 4 1**Explaination:** The middle element is 9 and if it is deleted this will be the stack traversal.

**Your Task:**  
You do not need to read input or print anything. Your task is to complete the function **deleteMid()** which takes n and st as input parameters and returns a stack where the middle element is deleted.

**Expected Time Complexity:** O(n)  
 **Expected Auxiliary Space:** O(n)

3. **Remove Duplicate from Sorted Array**

Given a sorted array **A[]** of size **N**, delete all the duplicated elements from A[]. Modify the array such that if there are X distinct elements in it then the first X positions of the array should be filled with them in increasing order and return the number of distinct elements in the array.  
   
Note:  
 1.Don't use set or HashMap to solve the problem.  
 2. You must return the number of distinct elements(X) in the array, the generated output will print all the elements of the modified array from index 0 to X-1.  
   
Example 1:

**Input:**N = 5Array = {2, 2, 2, 2, 2}**Output:** {2}**Explanation:** After removing all the duplicates only one instance of 2 will remain.

**Example 2:**

**Input:**N = 3Array = {1, 2, 2}**Output:** {1, 2}

**Your Task:**   
You don't need to read input or print anything. Complete the function **remove\_duplicate()** which takes the array A[] and its size N as input parameters and modifies it in place to delete all the duplicates. The function must return an integer X denoting the new modified size of the array.

**Expected Time Complexity:** O(N)  
 **Expected Auxiliary Space:** O(1)

**Constraints:**  
1 ≤ N ≤ 104  
1 ≤ A[i] ≤ 106

4.**Number Of Open Doors**

Consider a long alley with a **N** number of doors on one side. All the doors are closed initially. You move to and fro in the alley changing the states of the doors as follows: you open a door that is already closed and you close a door that is already opened. You start at one end go on altering the state of the doors till you reach the other end and then you come back and start altering the states of the doors again.  
 In the first go, you alter the states of doors numbered 1, 2, 3, , n.  
 In the second go, you alter the states of doors numbered 2, 4, 6  
 In the third go, you alter the states of doors numbered 3, 6, 9  
 You continue this till the Nth go in which you alter the state of the door numbered N.  
 You have to find the number of open doors at the end of the procedure.

**Example 1:**

**Input:N =** 2**Output:**1**Explanation:**Initially all doors are closed.After 1st go, all doors will be opened.After 2nd go second door will be closed.So, Only 1st door will remain Open.

**Example 2:**

**Input:N =** 4**Output:**2**Explanation:**Following the sequence 4 times, we cansee that only 1st and 4th doors willremain open.

**Your Task:**  
You don't need to read input or print anything. Your task is to complete the function **noOfOpenDoors()** which takes an Integer N as input and returns the answer.

**Expected Time Complexity:** O(√N)  
 **Expected Auxiliary Space:** O(1)

**Constraints:**  
1 <= N <= 1012

**5**. Given a number N. Find the last two digits of the Nth fibonacci number.  
 **Note:** If the last two digits are 02, return 2.

**Example 1:**

**Input:**N = 13**Output:**33**Explanation:**The 13th Fibonacci number is 233.So last two digits are 3 and 3.

**Example 2:**

**Input:**N = 255**Output:**70**Explanation:**The 255th fibonacci number is 875715953430-18854458033386304178158174356588264390370.Thus, last two digits are 7 and 0.

**Your Task:**  
You don't need to read input or print anything.Your task is to complete the function **fibonacciDigits()** which takes a number N as input parameter and returns the last two digits of the Nth fibonacci number.

**Expected Time Complexity:**O(K)  
 **Expected Auxillary Space:**O(K)  
 K is of the order 102.

**Constraints:**  
1<=N<=1018

**6.** Write a program that calculates the day of the week for any particular date in the past or future.

**Example 1:**

**Input:d =** 28, **m =** 12, **y =** 1995**Output:**Thursday**Explanation:**28 December 1995 was a Thursday.

**Example 2:**

**Input:d =** 30, **m =** 8, **y =** 2010**Output:**Monday**Explanation:**30 August 2010 was a Monday.

**Your Task:**  
You don't need to read input or print anything. Your task is to complete the function **getDayOfWeek()** which takes 3 Integers d, m and y denoting day, month and year as input and return a String denoting the answer.

**Expected Time Complexity:** O(1)  
 **Expected Auxiliary Space:** O(1)

**Constraints:**  
1 <= d <= 31  
 1 <= m <= 12  
 1990 <= y <= 2100

**7. Tower of Hanoi**

The [tower of Hanoi](https://en.wikipedia.org/wiki/Tower_of_Hanoi) is a famous puzzle where we have three rods and **N** disks. The objective of the puzzle is to move the entire stack to another rod. You are given the number of discs **N**. Initially, these discs are in the rod 1. You need to print all the steps of discs movement so that all the discs reach the 3rd rod. Also, you need to find the total moves.  
 **Note:** The discs are arranged such that the **top disc is numbered 1** and the **bottom-most disc is numbered N**. Also, all the discs have **different sizes** and a bigger disc **cannot** be put on the top of a smaller disc. Refer the provided link to get a better clarity about the puzzle.

**Example 1:**

**Input:**N = 2**Output:**move disk 1 from rod 1 to rod 2move disk 2 from rod 1 to rod 3move disk 1 from rod 2 to rod 33**Explanation:** For N=2 , steps will beas follows in the example and total3 steps will be taken.

**Example 2:**

**Input:**N = 3**Output:**move disk 1 from rod 1 to rod 3move disk 2 from rod 1 to rod 2move disk 1 from rod 3 to rod 2move disk 3 from rod 1 to rod 3move disk 1 from rod 2 to rod 1move disk 2 from rod 2 to rod 3move disk 1 from rod 1 to rod 37**Explanation:** For N=3 , steps will beas follows in the example and total7 steps will be taken.

**Your Task:**  
You don't need to read input or print anything. You only need to complete the function **toh()** that takes following parameters: **N** (number of discs), **from** (The rod number from which we move the disc), **to** (The rod number to which we move the disc), **aux** (The rod that is used as an auxiliary rod) and prints the required moves inside function body (See the example for the format of the output) as well as return the count of total moves made. The total number of moves are printed by the driver code.  
 **Please take care of the case of the letters.**

**Expected Time Complexity:** O(2N).  
 **Expected Auxiliary Space:** O(N).

**Constraints:**  
1 <= N <= 16

**8.** Given a sorted array of size **N** and an integer **K**, find the position at which **K** is present in the array using binary search.

**Example 1:**

**Input:**

N = 5

arr[] = {1 2 3 4 5}

K = 4

**Output:** 3

**Explanation:** 4 appears at index 3.

**Example 2:**

**Input:**

N = 5

arr[] = {11 22 33 44 55}

K = 445

**Output:** -1

**Explanation:** 445 is not present.

**Your Task:**   
You dont need to read input or print anything. Complete the function **binarysearch()** which takes **arr[]**, **N** and **K** as input parameters and returns the index of **K** in the array. If **K** is not present in the array, return -1.

**Expected Time Complexity:** O(LogN)  
 **Expected Auxiliary Space:** O(LogN) if solving recursively and O(1) otherwise.

**Constraints:**

* 1 <= N <= 105
* 1 <= arr[i] <= 106
* 1 <= K <= 106

**9. Anagram**

Given two strings **a** and **b** consisting of lowercase characters. The task is to check whether two given strings are an anagram of each other or not. An anagram of a string is another string that contains the same characters, only the order of characters can be different. For example, act and tac are an anagram of each other.

**Note** :- If the strings are anagrams you have to return True or else return False

**Example 1:**

**Input:**a = geeksforgeeks, b = forgeeksgeeks**Output:** YES**Explanation:** Both the string have same characters with same frequency. So, both are anagrams.

**Example 2:**

**Input:**a = allergy, b = allergic**Output:** NO**Explanation:** Characters in both the strings are not same, so they are not anagrams.

**Your Task:**  
You don't need to read input or print anything. Your task is to complete the function **isAnagram()** which takes the string **a** and string **b** as input parameter and check if the two strings are an anagram of each other. The function returns true if the strings are anagram else it returns false.

**Note: In python, you have to return True or False.**

**Expected Time Complexity:**O(|a|+|b|).  
 **Expected Auxiliary Space:**O(Number of distinct characters).

**Note:** |s| represents the length of string s.

**Constraints:**  
1 ≤ |a|,|b| ≤ 105

**10. Palindrome Pair :**

Given an array of strings arr[] of size N, find if there exists 2 strings arr[i] and arr[j] such that arr[i]+arr[j] is a palindrome i.e the concatenation of string arr[i] and arr[j] results into a palindrome.

**Example 1:**

**Input**:N = 6arr[] = {"geekf", "geeks", "or","keeg", "abc", "bc"}**Output:** 1 **Explanation**: There is a pair "geekf"and "keeg".

**Example 2:**

**Input:**N = 5arr[] = {"abc", "xyxcba", "geekst", "or", "bc"}**Output:** 1**Explanation**: There is a pair "abc"and "xyxcba".

**Your Task:**   
You don't need to read input or print anything. Your task is to complete the function **palindromepair()** which takes the array arr[], its size Nand returns true if palindrome pair exists and returns false otherwise.  
 The driver code itself prints 1 if returned value is true and prints 0 if returned value is false.

**Expected Time Complexity:** O(N\*l2) where l = length of longest string in arr[]  
 **Expected Auxiliary Space:** O(N\*l2) where l = length of longest string in arr[]

**Constraints:**  
1 ≤ N ≤ 2\*104  
1 ≤ |arr[i]| ≤ 10

**HARSHAVARDHANA M**

1. **Convert Array List To String Array In Java With Example.**

we have converted the whole list to array in three steps

a. Obtain the ArrayList size using size() method

b. Fetch each element of the list using get() method

c. Assigned each element to corresponding array element using assignment = operator

d. Print String array

**Output:**

Apple

Banana

Mango

1. **Problem**

**You are given a sorted array of integers. It is given that each element in the array is unique. You have to find the index where the element is located in the array. If it is not located in the array, you have to return the index at which it should be inserted in the array so that the array remains sorted.**

**You can’t use extra space and the expected time complexity is O(log2N) where N is the number of elements in the array. Since the array is sorted, we will use Binary Search to find the element. If the element is not found, the index at which we insert an element is always the ceil Index. So, what is the ceil index? At the end of the binary search, ceil index is where the low (or left) pointer points. So, the code for the same is shown below.**

**Sample Output When the element is present in the array Input: 4 1 3 5 6 5**

Output: 1 When the element is not present in the array

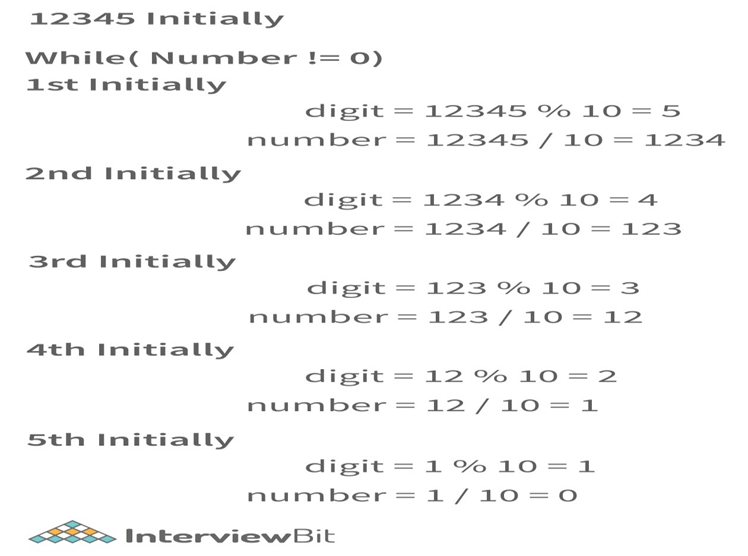
Input 2: 4 1 3 5 6 4

Output: 2

Time Complexity: The time complexity is O(log2N) where N is the number of elements in the array. Auxiliary Space: O (1) as we have not used any extra space

1. **Problem**

**Write a program in Java to count the digits in a number. Let us consider the number 12345. This is a 5 digit number. The number of digits.**



**In the image above, we have shown the way of extracting the digits of a number. However, in our questions, we just need to count the digits in the number. So, we have to count the number of times we can divide our input number by 10 before it becomes 0. Let us write the code based on the above algorithm. Output For Positive Number**

**Input: 1234**

**Output: The number of digits in 1234 is: 4 For**

**0 Input: 0**

**Output: The number of digits in 0 is: 1**

**For Negative Number Input: -12345**

**Output: The number of digits in -12345 is: 5 Corner Cases You Might Miss: We have used the loop and carried on iterations till the number becomes 0. What if the number was already 0? It still has 1 digit. So, we have handled that separately.**

**Also, to avoid any confusion, the negative numbers are converted to positive in our function and then we calculate their number of digits.**

**Time Complexity: O(log10N) where N is the input number. This is because we keep dividing the number by 10.**

**Auxiliary Space: O(1) as we have not used any extra space. Let us write the code based on the above algorithm.**

1. **Problem**

Write a program in Java to calculate the number of times a digit ‘D’ appears in a number N. You have to take N and D as inputs from the user.

This is the follow-up question to the previous question. In the previous question, we discussed how you can check the value of a digit using the modulus (%) operator. So, we will just use the previous code and in every iteration, we will check whether the digit is “D” or not. If it is D, increment the counter. The program for the same is shown below: Sample Input/Output.

Input: 142454 Output: The digit 4 occurs 3 times in 142454 Corner Cases You Might Miss: If the input number is 0 and the digit is also 0, it becomes a crucial corner case. This is because the number 0 has 1 frequency of digit 0 but it will not be handled correctly by our loop. So, we do this separately. Also, we have converted the negative numbers to positive ones to solve this problem.

Time Complexity: O(log10N) where N is the input number. This is because we keep dividing the number by 10.

Auxiliary Space: We have not used any auxiliary Space here. So, it is O(1).

1. **Problem**

**Write a program in Java to count the total number of vowels and consonants in a String.**

The string can contain all the alphanumeric and other special characters as well. However, only the lowercase English alphabets are allowed in the String. We just have to traverse the string. If we get any vowel (a,e,i,o,u), we increment the variable corresponding to the vowel count and if we get a consonant, we increment the variable corresponding to the consonant count

**Sample Output**

**Input: ae#zyu\***

**Output: The number of vowels is: 3 The number of consonants is: 2 The number of other characters is: 2**

**• Corner Cases, You Might Miss: In order to check whether a character is a vowel or not, we have a function. However, it is not right to say that if it is not a vowel then it will be a consonant as it can also be any other character. So, we have to make sure that it is an alphabet and then make sure that it is not a vowel. The same is done in the code.**

**• Time Complexity: O(N) where N is the length of the input string as we have to traverse the entire string once.**

Auxiliary Space: O(1) as we have not used any extra space.

1. **Problem**

**Write a program to print all the unique characters in a String. For instance, if the input string is “abcb”, the output will be the characters ‘a’ and ‘c’ as they are unique. The character ‘b’ repeats twice and so it will not be printed.**

**We can use a HashSet in order to store the characters of the String. When we arrive at a character in the String, if it is already present in the HashSet, we remove it from the HashSet as that character is not unique. If it is not present inside the HashSet, we add it to it. After traversing the entire string, we print the elements inside the HashMap.**

**Sample Output**

**Input: abcab**

**Output: c**

**Corner Cases, You Might Miss: What if such a string is passed that has all the duplicate characters? Also, it might happen that an empty string is passed as the input. So, in such a case, the size of the HashSet will remain 0 after processing the String. Hence, we have handled the case of a hashset having 0 sizes separately.**

**Time Complexity: The time complexity is O(N) where N is the length of the string as we are traversing the entire string. Auxiliary Space: O(N) as it might happen that all the N characters are unique so, we might generate a HashSet of size N.**

1. **Problem**

**Write a program in Java to prove that the strings are immutable in Java.**

Strings are immutable in Java. This can be proven by making changes to a string and comparing them with the == operator. Since this operator compares only the references, i.e. the addresses of the objects, it will be able to tell us if the changes are made to the same object or not. If after making changes to a string we compare it by == and we get no equals, this means that the strings are immutable.

**Output**

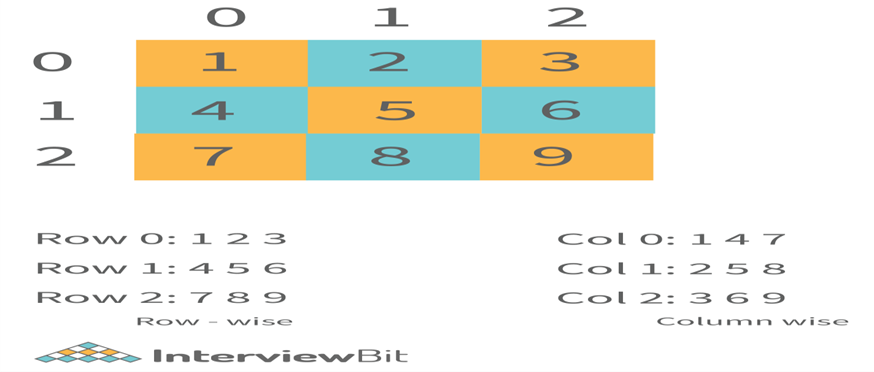
**True**

**False**

1. **Problem**

**Write a program in Java to input an NxN matrix and display it row-wise and column-wise Simply input the matrix.**

Now, display the matrix row-wise by starting from the first row and moving to the next elements within the same row. For displaying column-wise, start from the first column and keep moving in the same column to the next elements shown in below.



Input: 3 1 2 3 4 5 6 7 8 9

Output: Row 0 : 1 2 3

Row 1 : 4 5 6

Row 2 : 7 8 9

Col 0 : 1 4 7

Col 1 : 2 5 8

Col 2 : 3 6 9

• Time Complexity: O(N to the power 2) as we traverse the 2-D array to print it.

• Auxiliary Space: O(1) as we have not used any extra space.

1. **Problem**

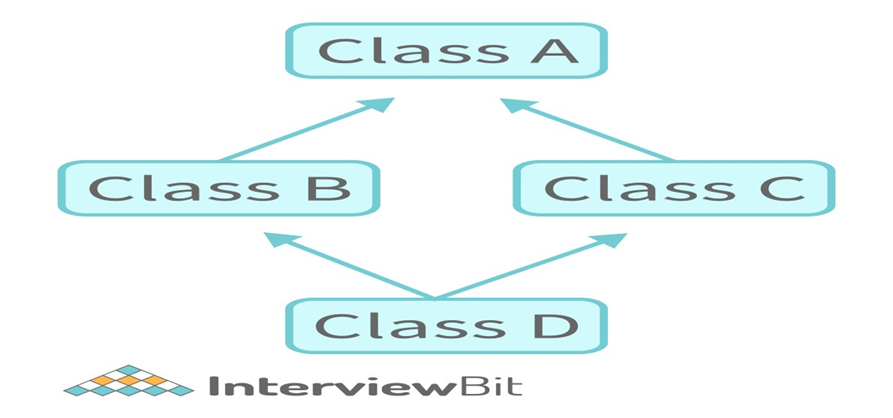
**Write a program in Java to show Thread Synchronization Here, you can create any 2 threads and synchronize them by using the synchronized keyword. An example program is given below.**

Output 5 10 15 20 25 30 35 40 45 50 6 12 18 24 30 36 42 48 54 60

1. **Problem**

**Write a program in Java to show the Diamond Problem.**

The Diamond Problem is a problem of Multiple inheritance. It is one of the major reasons why multiple inheritance is not supported in Java. Have a look at the diagram given below.



Here, class D extends from classes B and C and they extend from Class A. Let us say that class A has a function called print(). This function is overridden in Class B and C respectively. Now, when class D extends B and C both, say it calls super. print(). Which function should be called? This is an anomaly called the diamond problem or deadly diamond of death.

**Output**

This compilation error occurs because multiple inheritance is not allowed in Java.

Santosh Questions

1. Given a linked list, swap every two adjacent nodes and return its head. You must solve the problem without modifying the values in the list's nodes (i.e., only nodes themselves may be changed.)

Input: head = [1,2,3,4]

Output: [2,1,4,3]

1. Given two integers dividend and divisor, divide two integers without using multiplication, division, and mod operator.

The integer division should truncate toward zero, which means losing its fractional part. For example, 8.345 would be truncated to 8, and -2.7335 would be truncated to -2.

Return the quotient after dividing dividend by divisor.

Input: dividend = 10, divisor = 3

Output: 3

Explanation: 10/3 = 3.33333.. which is truncated to 3.

1. Given two non-negative integers num1 and num2 represented as strings, return the product of num1 and num2, also represented as a string.

Note: You must not use any built-in BigInteger library or convert the inputs to integer directly.

Example 1:

Input: num1 = "2", num2 = "3"

Output: "6"

1. You are given an n x n 2D matrix representing an image, rotate the image by 90 degrees (clockwise). You have to rotate the image in-place, which means you have to modify the input 2D matrix directly. DO NOT allocate another 2D matrix and do the rotation.

Input: matrix = [[1,2,3],[4,5,6],[7,8,9]]

Output: [[7,4,1],[8,5,2],[9,6,3]]

1. Given an unsorted integer array nums, return the smallest missing positive integer.You must implement an algorithm that runs in O(n) time and uses constant extra space.

Example 1:

Input: nums = [1,2,0]

Output: 3

Explanation: The numbers in the range [1,2] are all in the array.

Example 2:

Input: nums = [3,4,-1,1]

Output: 2

Explanation: 1 is in the array but 2 is missing.

Example 3:

Input: nums = [7,8,9,11,12]

Output: 1

Explanation: The smallest positive integer 1 is missing

1. Given a string containing just the characters '(' and ')', return the length of the longest valid (well-formed) parentheses

substring.

Example 1:

Input: s = "(()"

Output: 2

Explanation: The longest valid parentheses substring is "()".

Example 2:

Input: s = ")()())"

Output: 4

Explanation: The longest valid parentheses substring is "()()".

1. You are given an array of k linked-lists lists, each linked-list is sorted in ascending order. Merge all the linked-lists into one sorted linked-list and return it.

Input: lists = [[1,4,5],[1,3,4],[2,6]]

Output: [1,1,2,3,4,4,5,6]

Explanation: The linked-lists are:

[

1->4->5,

1->3->4,

2->6

]

merging them into one sorted list:

1->1->2->3->4->4->5->6

1. Given two binary strings a and b, return their sum as a binary string.

Example 1:

Input: a = "11", b = "1"

Output: "100"

Example 2:

Input: a = "1010", b = "1011"

Output: "10101"

1. Given an m x n grid of characters board and a string word, return true if word exists in the grid.

The word can be constructed from letters of sequentially adjacent cells, where adjacent cells are horizontally or vertically neighboring. The same letter cell may not be used more than once.

Input: board = [["A","B","C","E"],["S","F","C","S"],["A","D","E","E"]], word = "ABCCED"

Output: true

1. We can scramble a string s to get a string t using the following algorithm:

If the length of the string is 1, stop.

If the length of the string is > 1, do the following:

Split the string into two non-empty substrings at a random index, i.e., if the string is s, divide it to x and y where s = x + y.

Randomly decide to swap the two substrings or to keep them in the same order. i.e., after this step, s may become s = x + y or s = y + x.

Apply step 1 recursively on each of the two substrings x and y.

Given two strings s1 and s2 of the same length, return true if s2 is a scrambled string of s1, otherwise, return false.

Example 1:

Input: s1 = "abcde", s2 = "caebd"

Output: false

Example 2:

Input: s1 = "a", s2 = "a"

Output: true

Suvam Kumar

Question 1

Given an array of size N-1 such that it only contains distinct integers in the range of 1 to N. Find the missing element.

Example 1:

Input:N = 5

A[] = {1,2,3,5}Output: 4

2.Question

Write a program to print all the LEADERS in the array. An element is a leader if it is greater than all the elements to its right side. And the rightmost element is always a leader.

For example:

Input: arr[] = {16, 17, 4, 3, 5, 2},

Output: 17, 5, 2

Input: arr[] = {1, 2, 3, 4, 5, 2},

Output: 5, 2

3 Question

Given the heights of N towers and a value of K, Either increase or decrease the height of every tower by K (only once) where K > 0. After modifications, the task is to minimize the difference between the heights of the longest and the shortest tower and output its difference.

Examples:

Input: arr[] = {1, 15, 10}, k = 6

Output: Maximum difference is 5.

Explanation: Change 1 to 7, 15 to 9 and 10 to 4. Maximum difference is 5 (between 4 and 9). We can’t get a lower difference.

Input: arr[] = {1, 5, 15, 10}, k = 3

Output: Maximum difference is 8, arr[] = {4, 8, 12, 7}

4 Question

Given a Binary Tree, the task is to print the left view of the Binary Tree. The left view of a Binary Tree is a set of leftmost nodes for every level.

Examples:

Input:

4

/ \

5 2

/ \

3 1

/ \

6 7

Output: 4 5 3 6

5 Question

Given an array, print the Next Greater Element (NGE) for every element.

The Next greater Element for an element x is the first greater element on the right side of x in the array. Elements for which no greater element exist, consider the next greater element as -1.

Example:

Input: arr[] = [ 4 , 5 , 2 , 25 ]

Output: 4 –> 5

5 –> 25

2 –> 25

25 –> -1

Explanation: except 25 every element has an element greater than them present on the right side

Input: arr[] = [ 13 , 7, 6 , 12 ]

Output: 13 –> -1

7 –> 12

6 –> 12

12 –> -1

Explanation: 13 and 12 don’t have any element greater than them present on the right side

6 Question

Given a pointer to the head node of a linked list, the task is to reverse the linked list. We need to reverse the list by changing the links between nodes.

Examples:

Input: Head of following linked list

1->2->3->4->NULL

Output: Linked list should be changed to,

4->3->2->1->NULL

Input: Head of following linked list

1->2->3->4->5->NULL

Output: Linked list should be changed to,

5->4->3->2->1->NULL

7 Question

Given an array of integers, our task is to write a program that efficiently finds the second-largest element present in the array.

Example:

Input: arr[] = {12, 35, 1, 10, 34, 1}Output: The second largest element is 34.Explanation: The largest element of the

array is 35 and the second

largest element is 34

Input: arr[] = {10, 5, 10}Output: The second largest element is 5.Explanation: The largest element of

the array is 10 and the second

largest element is 5

Input: arr[] = {10, 10, 10}

Output: The second largest does not exist.Explanation: Largest element of the array

is 10 there is no second largest element

8 Question

Given an array of integers arr[] of size N and an integer, the task is to rotate the array elements to the left by d positions.

Examples:

Input:

arr[] = {1, 2, 3, 4, 5, 6, 7}, d = 2

Output: 3 4 5 6 7 1 2

Input: arr[] = {3, 4, 5, 6, 7, 1, 2}, d=2

Output: 5 6 7 1 2 3 4

9 Question

Given a string, recursively remove adjacent duplicate characters from the string. The output string should not have any adjacent duplicates. See the following examples.

Examples:

Input: azxxzy

Output: ay

• First “azxxzy” is reduced to “azzy”.

• The string “azzy” contains duplicates,

• so it is further reduced to “ay”.

Input: geeksforgeeg

Output: gksfor

10 Question

There is one meeting room in a firm. There are N meetings in the form of (S[i], F[i]) where S[i] is the start time of meeting i and F[i] is the finish time of meeting i. The task is to find the maximum number of meetings that can be accommodated in the meeting room. Print all meeting numbers

Examples:

Input: s[] = {1, 3, 0, 5, 8, 5}, f[] = {2, 4, 6, 7, 9, 9}

Output: 1 2 4 5

First meeting [1, 2]

Second meeting [3, 4]

Fourth meeting [5, 7]

Fifth meeting [8, 9]

Input : s[] = {75250, 50074, 43659, 8931, 11273, 27545, 50879, 77924},

f[] = {112960, 114515, 81825, 93424, 54316, 35533, 73383, 160252 }

Output : 6 7 1

Puneeth Kumar

1 Question

Problem

You are given an array A consisting of N integers. A subsequence of the array is called good if every pair of elements in the subsequence have an absolute difference of at most 10.

Task

Determine the maximum possible length of good subsequence.

Example

Assumptions

• N = 8

• A = [1, 9, 14, 2, 17, 14, 5, 18]

Approach

• Subsequence: [9,14,17,14,18] is good which is the maximum length subsequence.

• Hence the output is 5.

Function description

Complete the function solve() which takes the following 2 parameters and returns an integer as the required answer:

• N: Represents an integer denoting the size of the array

• A: Represents an array of N integers

Input format

Note: This is the input format you must use to provide custom input (available above the Compile and Test button).

• The first line contains an integer N.

• The second line contains an array A of N integers.

Output format

Print the maximum possible length of good subsequence.

Constraints

1 ≤ N ≤ 105

1 ≤ Ai ≤109

Sample Input

10

4 5 10 101 2 129 131 130 118 14

Sample Output

4

5

2 Question

Problem

You are developing the habit of "Safe Programming." You are given N unsigned integer data types of sizes (in bits) a1, a2, a3, ... , an. If the ith datatype has size ai bits, you can store all integers from 0 to 2ai -1.

The rule of safe programming is as follows:

• If n is a number that can be represented by the bit size ai, and if at least one aj > ai is present in the given array, then we must be able to represent n3 by any one of the bit sizes given in array a.

Task

Output 1 if it is "Safe Programming", else output 0.

Example

Assumptions

• N = 3

• A = [4, 3, 7]

Approach

The given bit sizes are 4, 3, and 7 bits.

Take n = 7. n can be represented by the data type of size a2 = 3 bits. { 23-1 = 7}

Here a1 = 4 and a3 = 7 are present in the given array such that a1 > a2 and a3 > a2.

So, according to the rule, we must be able to represent n3 = 343 using the given bit sizes i.e. 4 and 7 bits.

The maximum number that can be represented by the given sizes of bits is 2a3 - 1 = 27 - 1 = 127.

Clearly, 343 > 127. Hence we fail to represent 343 from the given bit sizes.

Hence answer will be 0.

Function description

Complete the SafeProgramming() function, which takes the following arguments, and returns true if it is safe programming, otherwise, returns false:

• N: Represents the number of available variations of bit sizes

• a[ ]: Represents the array of Available variations of bit sizes

Input format

• The first line contains N denoting the number of available variations of bit sizes

• The second line contains N space-separated integers denoting the available bit sizes.

Output Format

Output 1 if it is safe programming; else, output 0.

Constraints

2 ≤ N ≤ 1e6

3 ≤ a[i] ≤ 1e7

Sample Input

4

3 10 3 3

Sample Output

1

3 Question

Problem

You are given a string S containing lowercase English alphabets and a character k.

Task

Count the number of occurrences of k in S.

Example

Assumptions

• S = "abdbs"

• k = "b"

Approach

• You can see that the number of occurrences of b in S is 2. Hence, the answer is 2.

Function description

Complete the solve function which takes a string S and a character k as the argument and returns the number of occurrences of k in S:

• S: Represents the string

• k: Represents the character

Input format

Note: This is the input format that you must use to provide custom input (available above the Compile and Test button).

• The first line contains a single integer T, which denotes the number of test cases. T also specifies the number of times you have to run the solve function on a different set of inputs.

• For each test case:

o The first line contains a string S.

o The second line contains the character k.

Output format

For each test case, print a single line representing the number of occurrences of k in S.

Constraints

1≤T≤10

1≤|S|≤105

Sample Input

1

andadds

d

Sample Output

3

4 Question

Problem

You are given an array A of size N. You can apply the following operation any number of times (possibly zero) on A.

• Choose any two distinct indices X (0<=X<N) and Y (0<=Y<N) such that Bitwise Xor of A[X] and A[Y] is odd and swap A[X] and A[Y].

Find the lexicographically smallest array that can be obtained after performing the operation any number of times (possibly zero) on A.

An array P is lexicographically smaller than its permutation Q if and only if, for the earliest index at which P and Q differ, the element of P at that index is smaller than the element of Q at that index. Example, P = [1, 12, 4, 7, 8] is lexicographically smaller than Q = [1, 12, 8, 4, 7].

Input Format:

The first line contains an integer T, which denotes the number of test cases.

The first line of each test case contains an integer N, denoting the number of elements in the array A.

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

Output Format:

For each test case, print the lexicographically smallest array.

Constraints:

1<=T<=10

1<=N<=105

1<=A[i]<=105

Sample Input

3

4

2 1 4 3

4

1 2 16 15

2

1 1

Sample Output

1 2 3 4

1 2 15 16

1 1

5 Question

Problem

Given a string S consisting of lower case English alphabets and an integer K.

You need to form a new string P that follows the following conditions:

• Length of P should be equal to K + length of string S.

• P should only be formed by using the prefixes of string S.

• P should be lexicographically smallest.

Task

Determine the string P that satisfies all the given conditions.

Example

Assumptions

• N = 3

• K = 4

• S = bca

Approach

• Prefixes of S are "b", "bc", "bca".

• String P will be "bbbbbbb" of length 7 (N + K i.e. 3 + 4).

Function Description

Complete the solve function provided in the editor. This function takes the following 3 parameters and returns the valid string:

• N: Represents the size of string S.

• K: Represents the given integer.

• S: Represents the given string.

Input Format

Note: This is the input format you must use to provide custom input (available above the Compile and Test button).

• The first line contains a single integer T denoting the number of test cases.

• For each test case:

o The first line contains a single integer N denoting the size of string S.

o The second line contains a single integer K denoting the number of new characters added in string S.

o The third line contains the string S.

Output Format

For each test case in a separate line, output a string P that satisfies all the given conditions.

Constraints

1≤T≤102≤N≤2∗1052≤K≤2∗105

Code snippets (also called starter code/boilerplate code)

This question has code snippets for C, CPP, Java, and Python.

Sample Input

2

6

4

cbacaa

2

2

aa

Sample Output

cbacaacbac

aaaa

6 Question

Problem

You are given an integer N and your task is to make N a multiple of 3. In order to make N multiple of 3, you can insert at most one digit in N.

Your task is to find the minimum possible N which is a multiple of 3 after inserting at most one digit.

Note: You can insert the digit anywhere in N and also you need not necessarily insert.

Input format

• The first line contains an integer T denoting the number of test cases.

• The first line of each test case contains an integer N.

Output format

For each test case:

• Print the minimum N which is a multiple of 3 after at most one insertion.

Constraints

• 1≤T≤20000

• 1≤N≤109

Sample Input

1

5

Sample Output

15

Kishan Sakre P

1 Question

Define RESTful Root Resource Classes in the JAX-RS API?

2 Question

What do you understand by request method designator annotations?

3 Question

Write GET Method Example using URL

4 Question

Write POST Method Example Using Form

5 Question

Write HTTP Header Request Example

6 Question

Can I use comments inside a JSON file? If so, how?

7 Question

Explain the difference between JSON.stringify() and JSON.parse()

8 Question

Write JAXB Annotations

9 Question

Write Marshalling – Converting Java Object to XML

10 Question

1Unmarshalling – Converting XML to Java Object

Answer

var my\_object = {

key\_1: "some text",

key\_2: true,

key\_3: 5

};

var object\_as\_string = JSON.stringify(my\_object);

// "{"key\_1":"some text","key\_2":true,"key\_3":5}"

typeof(object\_as\_string);

// "string"

JSON.parse turns a string of JSON text into a JavaScript object, eg:

var object\_as\_string\_as\_object = JSON.parse(object\_as\_string);

// {key\_1: "some text", key\_2: true, key\_3: 5}

typeof(object\_as\_string\_as\_object);

// "object

When we run the code above, we can check the console output to verify that we have successfully converted XML data into a Java object:

Book [id=1, name=Book1, author=null, date=Sat Nov 12 11:38:18 ICT 2016]

Suvam Kumar

1 Answer

// Java code to implement the approach

import java.io.\*;

import java.util.\*;

class MissingNo {

// Function to find the missing number

public static void findMissing(int arr[], int N)

{

int i;

int temp[] = new int[N + 1];

for (i = 0; i <= N; i++) {

temp[i] = 0;

}

for (i = 0; i < N; i++) {

temp[arr[i] - 1] = 1;

}

int ans = 0;

for (i = 0; i <= N; i++) {

if (temp[i] == 0)

ans = i + 1;

}

System.out.println(ans);

}

// Driver Code

public static void main(String[] args)

{

int arr[] = { 1, 3, 7, 5, 6, 2 };

int n = arr.length;

// Function call

findMissing(arr, n);

}

}

2 Answer

// Java program to sort an array of 0, 1 and 2

import java.io.\*;

class countzot {

// Sort the input array, the array is assumed to

// have values in {0, 1, 2}

static void sort012(int a[], int arr\_size)

{

int lo = 0;

int hi = arr\_size - 1;

int mid = 0, temp = 0;

// Iterate till all the elements

// are sorted

while (mid <= hi) {

switch (a[mid]) {

// If the element is 0

case 0: {

temp = a[lo];

a[lo] = a[mid];

a[mid] = temp;

lo++;

mid++;

break;

}

// If the element is 1

case 1:

mid++;

break;

// If the element is 2

case 2: {

temp = a[mid];

a[mid] = a[hi];

a[hi] = temp;

hi--;

break;

}

}

}

}

/\* Utility function to print array arr[] \*/

static void printArray(int arr[], int arr\_size)

{

int i;

for (i = 0; i < arr\_size; i++)

System.out.print(arr[i] + " ");

System.out.println("");

}

/\*Driver function to check for above functions\*/

public static void main(String[] args)

{

int arr[] = { 0, 1, 1, 0, 1, 2, 1, 2, 0, 0, 0, 1 };

int arr\_size = arr.length;

sort012(arr, arr\_size);

printArray(arr, arr\_size);

}

}

3 Answer

/\*package whatever //do not write package name here \*/

import java.io.\*;

import java.util.\*;

// Driver code

public class Main {

public static void main(String[] args)

{

int[] arr = { 7, 4, 8, 8, 8, 9 };

int k = 6;

int ans = getMinDiff(arr, arr.length, k);

System.out.println(ans);

}

// User function Template for Java

public static int getMinDiff(int[] arr, int n, int k)

{

Arrays.sort(arr);

// Maximum possible height difference

int ans = arr[n - 1] - arr[0];

int tempmin, tempmax;

tempmin = arr[0];

tempmax = arr[n - 1];

for (int i = 1; i < n; i++) {

// if on subtracting k we got negative then

// continue

if (arr[i] - k < 0)

continue;

// Minimum element when we add k to whole array

tempmin = Math.min(arr[0] + k, arr[i] - k);

// Maximum element when we subtract k from whole

// array

tempmax

= Math.max(arr[i - 1] + k, arr[n - 1] - k);

ans = Math.min(ans, tempmax - tempmin);

}

return ans;

}

}

4 Answer

// Java program to print left view of binary tree

/\* Class containing left and right child of current

node and key value\*/

class Node {

int data;

Node left, right;

public Node(int item)

{

data = item;

left = right = null;

}

}

/\* Class to print the left view \*/

class BinaryTree {

Node root;

static int max\_level = 0;

// recursive function to print left view

void leftViewUtil(Node node, int level)

{

// Base Case

if (node == null)

return;

// If this is the first node of its level

if (max\_level < level) {

System.out.print(node.data + " ");

max\_level = level;

}

// Recur for left and right subtrees

leftViewUtil(node.left, level + 1);

leftViewUtil(node.right, level + 1);

}

// A wrapper over leftViewUtil()

void leftView()

{

max\_level = 0;

leftViewUtil(root, 1);

}

/\* testing for example nodes \*/

public static void main(String args[])

{

/\* creating a binary tree and entering the nodes \*/

BinaryTree tree = new BinaryTree();

tree.root = new Node(10);

tree.root.left = new Node(2);

tree.root.right = new Node(3);

tree.root.left.left = new Node(7);

tree.root.left.right = new Node(8);

tree.root.right.right = new Node(15);

tree.root.right.left = new Node(12);

tree.root.right.right.left = new Node(14);

tree.leftView();

}

}

5 Answer

// Simple Java program to print next

// greater elements in a given array

class Main {

/\* prints element and NGE pair for

all elements of arr[] of size n \*/

static void printNGE(int arr[], int n)

{

int next, i, j;

for (i = 0; i < n; i++) {

next = -1;

for (j = i + 1; j < n; j++) {

if (arr[i] < arr[j]) {

next = arr[j];

break;

}

}

System.out.println(arr[i] + " -- " + next);

}

}

public static void main(String args[])

{

int arr[] = { 11, 13, 21, 3 };

int n = arr.length;

printNGE(arr, n);

}

}

6 Answer

// Java program for reversing the linked list

class LinkedList {

static Node head;

static class Node {

int data;

Node next;

Node(int d)

{

data = d;

next = null;

}

}

/\* Function to reverse the linked list \*/

Node reverse(Node node)

{

Node prev = null;

Node current = node;

Node next = null;

while (current != null) {

next = current.next;

current.next = prev;

prev = current;

current = next;

}

node = prev;

return node;

}

// prints content of double linked list

void printList(Node node)

{

while (node != null) {

System.out.print(node.data + " ");

node = node.next;

}

}

// Driver Code

public static void main(String[] args)

{

LinkedList list = new LinkedList();

list.head = new Node(85);

list.head.next = new Node(15);

list.head.next.next = new Node(4);

list.head.next.next.next = new Node(20);

System.out.println("Given linked list");

list.printList(head);

head = list.reverse(head);

System.out.println("");

System.out.println("Reversed linked list ");

list.printList(head);

}

}

7 Answer

// Java program to find second largest

// element in an array

import java.util.\*;

class GFG{

// Function to print the

// second largest elements

static void print2largest(int arr[], int arr\_size)

{

int i, first, second;

// There should be

// atleast two elements

if (arr\_size < 2)

{

System.out.printf(" Invalid Input ");

return;

}

// Sort the array

Arrays.sort(arr);

// Start from second last element

// as the largest element is at last

for (i = arr\_size - 2; i >= 0; i--)

{

// If the element is not

// equal to largest element

if (arr[i] != arr[arr\_size - 1])

{

System.out.printf("The second largest " +"element is %d\n", arr[i]);

return;

}

}

System.out.printf("There is no second " + "largest element\n");

}

// Driver code

public static void main(String[] args)

{

int arr[] = {12, 35, 1, 10, 34, 1};

int n = arr.length;

print2largest(arr, n);

}

}

8 Answer

/\*package whatever //do not write package name here \*/

import java.io.\*;

class GFG {

// Function to rotate array

static void Rotate(int arr[], int d, int n)

{

// Storing rotated version of array

int temp[] = new int[n];

// Keeping track of the current index

// of temp[]

int k = 0;

// Storing the n - d elements of

// array arr[] to the front of temp[]

for (int i = d; i < n; i++) {

temp[k] = arr[i];

k++;

}

// Storing the first d elements of array arr[]

// into temp

for (int i = 0; i < d; i++) {

temp[k] = arr[i];

k++;

}

// Copying the elements of temp[] in arr[]

// to get the final rotated array

for (int i = 0; i < n; i++) {

arr[i] = temp[i];

}

}

// Function to print elements of array

static void PrintTheArray(int arr[], int n)

{

for (int i = 0; i < n; i++) {

System.out.print(arr[i]+" ");

}

}

public static void main (String[] args) {

int arr[] = { 1, 2, 3, 4, 5, 6, 7 };

int N = arr.length;

int d = 2;

// Function calling

Rotate(arr, d, N);

PrintTheArray(arr, N);

}

}

9 Answer

// Java program to remove all adjacent duplicates from a

// string

import java.io.\*;

import java.util.\*;

class GFG {

static char last\_removed; // will store the last char

// removed during recursion

// Recursively removes adjacent duplicates from str and

// returns new string. last\_removed is a pointer to

// last\_removed character

static String removeUtil(String str)

{

// If length of string is 1 or 0

if (str.length() == 0 || str.length() == 1)

return str;

// Remove leftmost same characters and recur for

// remaining string

if (str.charAt(0) == str.charAt(1)) {

last\_removed = str.charAt(0);

while (str.length() > 1

&& str.charAt(0) == str.charAt(1))

str = str.substring(1, str.length());

str = str.substring(1, str.length());

return removeUtil(str);

}

// At this point, the first character is definitely

// different from its adjacent. Ignore first

// character and recursively remove characters from

// remaining string

String rem\_str

= removeUtil(str.substring(1, str.length()));

// Check if the first character of the rem\_string

// matches with the first character of the original

// string

if (rem\_str.length() != 0

&& rem\_str.charAt(0) == str.charAt(0)) {

last\_removed = str.charAt(0);

// Remove first character

return rem\_str.substring(1, rem\_str.length());

}

// If remaining string becomes empty and last

// removed character is same as first character of

// original string. This is needed for a string like

// "acbbcddc"

if (rem\_str.length() == 0

&& last\_removed == str.charAt(0))

return rem\_str;

// If the two first characters of str and rem\_str

// don't match, append first character of str before

// the first character of rem\_str

return (str.charAt(0) + rem\_str);

}

static String remove(String str)

{

last\_removed = '\0';

return removeUtil(str);

}

// Driver code

public static void main(String args[])

{

String str1 = "geeksforgeeg";

System.out.println(remove(str1));

String str2 = "azxxxzy";

System.out.println(remove(str2));

String str3 = "caaabbbaac";

System.out.println(remove(str3));

String str4 = "gghhg";

System.out.println(remove(str4));

String str5 = "aaaacddddcappp";

System.out.println(remove(str5));

String str6 = "aaaaaaaaaa";

System.out.println(remove(str6));

String str7 = "qpaaaaadaaaaadprq";

System.out.println(remove(str7));

String str8 = "acaaabbbacdddd";

System.out.println(remove(str8));

}

}

10 Answer

// Java program to find maximum number of meetings

import java.util.\*;

// Comparator function which can compare

// the ending time of the meeting ans

// sort the list

class mycomparator implements Comparator<meeting> {

@Override public int compare(meeting o1, meeting o2)

{

if (o1.end < o2.end) {

// Return -1 if second object is

// bigger than first

return -1;

}

else if (o1.end > o2.end)

// Return 1 if second object is

// smaller than first

return 1;

return 0;

}

}

// Custom class for storing starting time,

// finishing time and position of meeting.

class meeting {

int start;

int end;

int pos;

meeting(int start, int end, int pos)

{

this.start = start;

this.end = end;

this.pos = pos;

}

}

class GFG {

// Function for finding maximum meeting in one room

public static void maxMeeting(ArrayList<meeting> al, int s)

{

// Initialising an arraylist for storing answer

ArrayList<Integer> m = new ArrayList<>();

int time\_limit = 0;

mycomparator mc = new mycomparator();

// Sorting of meeting according to

// their finish time.

Collections.sort(al, mc);

// Initially select first meeting.

m.add(al.get(0).pos);

// time\_limit to check whether new

// meeting can be conducted or not.

time\_limit = al.get(0).end;

// Check for all meeting whether it

// can be selected or not.

for (int i = 1; i < al.size(); i++) {

if (al.get(i).start > time\_limit) {

// Add selected meeting to arraylist

m.add(al.get(i).pos);

// Update time limit

time\_limit = al.get(i).end;

}

}

// Print final selected meetings.

for (int i = 0; i < m.size(); i++)

System.out.print(m.get(i) + 1 + " ");

}

// Driver's Code

public static void main(String[] args)

{

// Starting time

int s[] = { 1, 3, 0, 5, 8, 5 };

// Finish time

int f[] = { 2, 4, 6, 7, 9, 9 };

// Defining an arraylist of meet type

ArrayList<meeting> meet = new ArrayList<>();

for (int i = 0; i < s.length; i++)

// Creating object of meeting

// and adding in the list

meet.add(new meeting(s[i], f[i], i));

// Function call

maxMeeting(meet, meet.size());

}

}

Kishan Sakre P

1 Answer

• A resource class is nothing but a Java class that uses JAX-RS provided annotations for implementing web resources.

• They are the POJOs that are annotated either with @Path or have at least one method annotated with @Path, @GET, @POST, @DELETE, @PUT, etc.

import javax.ws.rs.Path;

/\*\*

\* InterviewBitService is a root resource class that is exposed at 'resource\_service' path

\*/

@Path('resource\_service')

public class InterviewBitService {

// Defined methods

}

2 Answer

They are the runtime annotations in the JAX-RS library that are applied to Java methods. They correspond to the HTTP request methods that the clients want to make. They are @GET, @POST, @PUT, @DELETE, @HEAD.

import javax.ws.rs.Path;

/\*\*

\* InterviewBitService is a root resource class that is exposed at 'resource\_service' path

\*/

@Path('resource\_service')

public class InterviewBitService {

@GET

public String getRESTQuestions() {

// some operations

}

}

3 Answer

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

// Extend HttpServlet class

public class HelloForm extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Set response content type

response.setContentType("text/html");

PrintWriter out = response.getWriter();

String title = "Using GET Method to Read Form Data";

String docType =

"<!doctype html public \"-//w3c//dtd html 4.0 " + "transitional//en\">\n";

out.println(docType +

"<html>\n" +

"<head><title>" + title + "</title></head>\n" +

"<body bgcolor = \"#f0f0f0\">\n" +

"<h1 align = \"center\">" + title + "</h1>\n" +

"<ul>\n" +

" <li><b>First Name</b>: "

+ request.getParameter("first\_name") + "\n" +

" <li><b>Last Name</b>: "

+ request.getParameter("last\_name") + "\n" +

"</ul>\n" +

"</body>" +

"</html>"

);

}

}

4 Answer

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

// Extend HttpServlet class

public class HelloForm extends HttpServlet {

// Method to handle GET method request.

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Set response content type

response.setContentType("text/html");

PrintWriter out = response.getWriter();

String title = "Using GET Method to Read Form Data";

String docType =

"<!doctype html public \"-//w3c//dtd html 4.0 " +

"transitional//en\">\n";

out.println(docType +

"<html>\n" +

"<head><title>" + title + "</title></head>\n" +

"<body bgcolor = \"#f0f0f0\">\n" +

"<h1 align = \"center\">" + title + "</h1>\n" +

"<ul>\n" +

" <li><b>First Name</b>: "

+ request.getParameter("first\_name") + "\n" +

" <li><b>Last Name</b>: "

+ request.getParameter("last\_name") + "\n" +

"</ul>\n" +

"</body>"

"</html>"

);

}

// Method to handle POST method request.

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doGet(request, response);

}

}

5 Answer

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.util.\*;

// Extend HttpServlet class

public class DisplayHeader extends HttpServlet {

// Method to handle GET method request.

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Set response content type

response.setContentType("text/html");

PrintWriter out = response.getWriter();

String title = "HTTP Header Request Example";

String docType =

"<!doctype html public \"-//w3c//dtd html 4.0 " + "transitional//en\">\n";

out.println(docType +

"<html>\n" +

"<head><title>" + title + "</title></head>\n"+

"<body bgcolor = \"#f0f0f0\">\n" +

"<h1 align = \"center\">" + title + "</h1>\n" +

"<table width = \"100%\" border = \"1\" align = \"center\">\n" +

"<tr bgcolor = \"#949494\">\n" +

"<th>Header Name</th><th>Header Value(s)</th>\n"+

"</tr>\n"

);

Enumeration headerNames = request.getHeaderNames();

while(headerNames.hasMoreElements()) {

String paramName = (String)headerNames.nextElement();

out.print("<tr><td>" + paramName + "</td>\n");

String paramValue = request.getHeader(paramName);

out.println("<td> " + paramValue + "</td></tr>\n");

}

out.println("</table>\n</body></html>");

}

// Method to handle POST method request.

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doGet(request, response);

}

}

6 Answer

No.

The JSON should all be data, and if you include a comment, then it will be data too.

You could have a designated data element called "\_comment" (or something) that would be ignored by apps that use the JSON data.

{

"\_comment": "comment text goes here...",

"glossary": {

"title": "example glossary",

"GlossDiv": {

"title": "S",

"GlossList": {

"GlossEntry": {

"ID": "SGML",

"SortAs": "SGML",

"GlossTerm": "Standard Generalized Markup Language",

"Acronym": "SGML",

"Abbrev": "ISO 8879:1986",

"GlossDef": {

"para": "A meta-markup language, used to create markup languages such as DocBook.",

"GlossSeeAlso": ["GML", "XML"]

},

"GlossSee": "markup"

}

}

}

}

}

7 Answer

JSON.stringify turns a JavaScript object into JSON text and stores that JSON text in a string, eg:

var my\_object = {

key\_1: "some text",

key\_2: true,

key\_3:

};

var object\_as\_string = JSON.stringify(my\_object)

// "{"key\_1":"some text","key\_2":true,"key\_3":5}"

typeof(object\_as\_string);

// "string"

JSON.parse turns a string of JSON text into a JavaScript object, eg:

var object\_as\_string\_as\_object = JSON.parse(object\_as\_string);

// {key\_1: "some text", key\_2: true, key\_3: 5}

typeof(object\_as\_string\_as\_object);

// "object

8 Answer

JAXB uses Java annotations for augmenting the generated classes with additional information. Adding such annotations to existing Java classes prepares them for the JAXB runtime.

@XmlRootElement(name = "book")

@XmlType(propOrder = { "id", "name", "date" })

public class Book {

private Long id;

private String name;

private String author;

private Date date;

@XmlAttribute

public void setId(Long id) {

this.id = id;

}

@XmlElement(name = "title")

public void setName(String name) {

this.name = name;

}

@XmlTransient

public void setAuthor(String author) {

this.author = author;

} // constructor, getters and setters

}

9 Answer

Marshalling gives a client application the ability to convert a JAXB-derived Java object tree into XML data. By default, the Marshaller uses UTF-8 encoding when generating XML data. Next, we will generate XML files from Java objects.

public void marshal() throws JAXBException, IOException {

Book book = new Book();

book.setId(1L);

book.setName("Book1");

book.setAuthor("Author1");

book.setDate(new Date());

JAXBContext context = JAXBContext.newInstance(Book.class); Marshaller mar= context.createMarshaller(); mar.setProperty(Marshaller.JAXB\_FORMATTED\_OUTPUT, Boolean.TRUE); mar.marshal(book, new File("./book.xml"));

}

The javax.xml.bind.JAXBContext class provides a client's entry point to JAXB API. By default, JAXB does not format the XML document. This saves space and prevents that any whitespace is accidentally interpreted as significant.

To have JAXB format the output, we simply set the Marshaller.JAXB\_FORMATTED\_OUTPUT property to true on the Marshaller. The marshal method uses an object and an output file to store the generated XML as parameters.

When we run the code above, we can check the result in the book.xml to verify that we have successfully converted a Java object into XML data:

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<book id="1">

<title>Book1</title>

<date>2016-11-12T11:25:12.227+07:00</date>

</book>

10 Answer

var my\_object = {

key\_1: "some text",

key\_2: true,

key\_3: 5

};

var object\_as\_string = JSON.stringify(my\_object);

// "{"key\_1":"some text","key\_2":true,"key\_3":5}"

typeof(object\_as\_string);

// "string"

JSON.parse turns a string of JSON text into a JavaScript object, eg:

var object\_as\_string\_as\_object = JSON.parse(object\_as\_string);

// {key\_1: "some text", key\_2: true, key\_3: 5}

typeof(object\_as\_string\_as\_object);

// "object

When we run the code above, we can check the console output to verify that we have successfully converted XML data into a Java object:

Book [id=1, name=Book1, author=null, date=Sat Nov 12 11:38:18 ICT 2016]

Ghanavi Answer

1.Answer

import java.util.Arrays;

import java.util.TreeSet;

import java.util.\*;

import java.io.\*;

import static java.util.stream.Collectors.toCollection;

class Main {

public static String FindIntersection(String[] strArr) {

TreeSet<Integer> set1 = Arrays.stream(strArr[0].replace(" ", "").split(",")).map(Integer::parseInt).collect(toCollection(TreeSet::new));

TreeSet<Integer> set2 = Arrays.stream(strArr[1].replace(" ", "").split(",")).map(Integer::parseInt).collect(toCollection(TreeSet::new));

set1.retainAll(set2);

return (set1.isEmpty() ? "false" : Arrays.toString(new TreeSet<>(set1).toArray()).replace("]", "").replace("[", "").replace(" ", ""));

}

public static void main (String[] args) {

// keep this function call here

Scanner s = new Scanner(System.in);

System.out.print(FindIntersection(s.nextLine()));

}

}

2.Answer

// our two sum function which will return

// all pairs in the array that sum up to S

function twoSum(arr, S) {

var sums = [];

// check each element in array

for (var i = 0; i < arr.length; i++) {

// check each other element in the array

for (var j = i + 1; j < arr.length; j++) {

// determine if these two elements sum to S

if (arr[i] + arr[j] === S) {

sums.push([arr[i], arr[j]]);

}

}

}

// return all pairs of integers that sum to S

return sums;

}

twoSum([3, 5, 2, -4, 8, 11], 7);

}

3.Anwer

import java.util.\*;

import java.io.\*;

class Function {

String LongestWord(String sen) {

String[] senArray = sen.split("[^a-z^0-9]");

int longestIndex = 0;

String longestWord = "";

for (int i = 0; i < senArray.length; i++) {

//System.out.println(senArray[i]);

if (senArray[i].length() > longestIndex) {

longestWord = senArray[i];

longestIndex = senArray[i].length();

}

}

return longestWord;

}

public static void main (String[] args) {

// keep this function call here

Scanner s = new Scanner(System.in);

Function c = new Function();

System.out.print(c.LongestWord(s.nextLine()));

}

}

4.Answer

import java.util.\*;

import java.io.\*;

class Main {

public static String QuestionsMarks(String str) {

int i = 0;

while(i < str.length()) {

if(Character.isDigit(str.charAt(i))) {

int j = i + 1;

int qCount = 0;

while(j < str.length()) {

if(str.charAt(j) == '?') qCount++;

if(Character.isDigit(str.charAt(j))) {

int sum = Integer.parseInt(str.charAt(i) + "") + Integer.parseInt(str.charAt(j) + "");

if(!(sum == 10 && qCount == 3)) return "false";

i = j;

break;

}

j++;

}

}

i++;

}

return "true";

}

public static void main (String[] args) {

// keep this function call here

Scanner s = new Scanner(System.in);

System.out.print(QuestionsMarks(s.nextLine()));

}

}

5.Answer

import java.util.\*;

import java.lang.\*;

import java.io.\*;

class Codechef

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

Scanner sc = new Scanner(System.in);

int T = sc.nextInt();

while(T-->0)

{

int n=sc.nextInt();

int arr[] = new int[n];

for(int i=0;i<n;i++){

arr[i]=sc.nextInt();

}

Arrays.sort(arr);

System.out.println(arr[1]);

}

}

}

6.Answer

import java.util.\*;

import java.lang.\*;

import java.io.\*;

class Codechef

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

Scanner sc = new Scanner(System.in);

int T = sc.nextInt();

while(T-->0)

{

long a=sc.nextLong();

long b=sc.nextLong();

long c=sc.nextLong();

if(a==b && b==c) System.out.println("NO");

else if(a+b==c) System.out.println("YES");

else if(b+c==a) System.out.println("YES");

else if(c+a==b) System.out.println("YES");

else System.out.println("NO");

}

}

}

7.Answer

import java.util.\*;

import java.lang.\*;

import java.io.\*;

class Codechef

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

while(n-->0){

char s[] = sc.next().toCharArray();

int a = s.length;

long c =0;

long r=0;

for(int i=0; i<a; i++){

if(s[i]=='7' && c>=2){

r += ((c\*(c-1))/2);

}else if(s[i]=='0'){

c++;

}

}

System.out.println(r);

}

}

}

8.Answer

import java.util.\*;

import java.lang.\*;

import java.io.\*;

class Codechef

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

while(n-->0){

int a = sc.nextInt();

int []arr = new int[a];

for(int i=0; i<a; i++){

arr[i] = sc.nextInt();

}

int peak =1; int maxpeak=1;

int insc=0;

for(int i=1; i<a; i++){

if(arr[i-1]<arr[i]){

peak = insc==1 ? peak+1 : 2;

insc = 1;

}else if(arr[i]==arr[i-1]){

peak=0;

insc=0;

}

else {

peak = (insc != 0) ? (peak+1) : 2;

insc = -1;

}

if(maxpeak < peak)

maxpeak = peak;

}

System.out.println(maxpeak);

}

}

}

9.Answer

import java.util.\*;

import java.lang.\*;

import java.io.\*;

class Codechef

{

public static void main (String[] args) throws java.lang.Exception

{

Scanner sc=new Scanner(System.in);

int t=sc.nextInt();

for(int j=0;j<t;j++){

long n=sc.nextLong();

long ans=1;

for(int i=2;i\*i<=n;i++){

int c=0;

while(n%i==0){

n=n/i;

c++;

}

if(c>0){

ans=ans\*i;

}

}

if(n>2)ans=ans\*n;

System.out.println(ans);

}

}

}

10.Answer

/\* package codechef; // don't place package name! \*/

import java.util.\*;

import java.lang.\*;

import java.io.\*;

/\* Name of the class has to be "Main" only if the class is public. \*/

class Codechef

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

Scanner sc = new Scanner(System.in);

int t = sc.nextInt();

while(t-- >0){

int n = sc.nextInt();

String s = sc.next();

int count= 0;

int temp = 0;

for(int i=0; i<s.length(); i++){

if(s.charAt(i)=='('){

count++;

}

else if(s.charAt(i)== ')'){

count--;

}

if(count <0){

temp++;

count=0;

}

}

System.out.println(temp+count);

}

}

}

Answer Dhanyashree

1.Simple

class Codechef

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

try{

Scanner in=new Scanner(System.in);

int t=in.nextInt();

while(t>0)

{

long h = in.nextLong();

long x = in.nextLong();

long y = in.nextLong();

if (x>y)

System.out.println(1);

else

System.out.println(0);

t--;

}

}catch(Exception e){

return;

}

}

}

### 2. Intermediate

class Codechef

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

// your code goes here

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int []a=new int[n];

int c=0;

int m=0;

for(int i=0;i<n;i++)

a[i]=sc.nextInt();

for(int i=0;i<n;i++)

{

if(a[i]>m)

{c++;

m=a[i];}

}

System.out.println(c);

}

}

3. Intermediate

class FastScanner

{

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

StringTokenizer st;

String next()

{

while(st==null || !st.hasMoreTokens())

{

try{

st=new StringTokenizer(br.readLine());

}

catch(IOException e){}

}

return st.nextToken();

}

int nextInt()

{

return Integer.parseInt(next());

}

float nextFloat()

{

return Float.parseFloat(next());

}

long nextLong()

{

return Long.parseLong(next());

}

double nextDouble()

{

return Double.parseDouble(next());

}

String nextLine()

{

String s="";

try{

s=br.readLine();

}

catch(IOException e){}

return s;

}

}

class codechef{

public static void main (String[] args) throws java.lang.Exception

{

FastScanner input = new FastScanner();

StringBuilder sb = new StringBuilder();

int t = input.nextInt();

while(t-- > 0) {

int n = input.nextInt();

ArrayList<Integer> oneIndexes = new ArrayList<Integer>();

for(int i = 0; i < n; i++) {

int a = input.nextInt();

if(a == 1) oneIndexes.add(i+1);

}

int[] p = new int[n+1];

int index = 1;

int prevValue = 0;

for(int val : oneIndexes) {

int temp = val;

while(temp > prevValue)

p[index++] = temp--;

prevValue = val;

}

for(int i = 1; i <= n; i++)

sb.append(p[i] + " ");

sb.append("\n");

}

System.out.print(sb);

System.out.flush();

}

}

4.Simple

class Codechef

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

Scanner ss=new Scanner(System.in);

int t=ss.nextInt();

for(int a=0;a<t;a++)

{

int x=ss.nextInt();

int num=x+1;

String number="";

boolean status=false, found=false;

int count=0;

while(true)

{

number=num+"";

for(int f=0;f<number.length();f++)

{

for(int g=0;g<number.length();g++)

{

if(f!=g)

{

if(number.charAt(f)==number.charAt(g))

{

count++;

}

}

}

}

if(count==0)

{

// System.out.println("Hello"+num);

break;

}

else

{

// System.out.println(num);

count=0;

num++;

}

}

System.out.println(num);

}

}

}

5.Array intermediate

class Codechef

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

Scanner sc = new Scanner(System.in);

int T = sc.nextInt();

while(T-->0){

ArrayList<Integer>arrlist1=new ArrayList<Integer>();

int N = sc.nextInt();

for(int i=0;i<N;i++){

arrlist1.add(sc.nextInt());

}

int max=0;

int count=0;

for(int i=0;i<arrlist1.size();i++){

max=Math.max(arrlist1.get(i),max);

if(max==i+1){

count++;

}

}

System.out.println(count);

}

}

}

6.Array complex

import java.util.\*;

import java.io.\*;

class Main {

static FastReader scan = new FastReader();

static int pow\_of\_two(int num) {

int ans = 0;

for (; num % 2 == 0; num >>= 1, ans++);

return ans;

}

static void print\_arr(int[] arr) {

for (int i = 0; i < arr.length; i++) {

System.out.print(arr[i] + " ");

}

System.out.println();

}

public static void main(String[] args) {

int N = scan.nextInt(), M = scan.nextInt();

int[] arr = new int[N];

for(int i = 0; i < N; i++) {

arr[i] = pow\_of\_two(scan.nextInt());

}

int[] diff = new int[N];

// print\_arr(arr);

for (int i = 0; i < M; i++) {

int l = scan.nextInt();

int r = scan.nextInt();

int val = pow\_of\_two(scan.nextInt());

l--; r--;

diff[l] += val;

if (r + 1 != N) {

diff[r + 1] -= val;

}

// print\_arr(diff);

}

for (int i = 1; i < N; i++) {

diff[i] += diff[i - 1];

}

// print\_arr(arr);

int ans = 0;

for (int i = 0; i < N; i++) {

ans = Math.max(ans, arr[i] + diff[i]);

}

System.out.println(ans);

}

}

class FastReader {

BufferedReader br;

StringTokenizer st;

public FastReader()

{

br = new BufferedReader(

new InputStreamReader(System.in));

}

String next()

{

while (st == null || !st.hasMoreElements()) {

try {

st = new StringTokenizer(br.readLine());

}

catch (IOException e) {

e.printStackTrace();

}

}

return st.nextToken();

}

int nextInt() { return Integer.parseInt(next()); }

long nextLong() { return Long.parseLong(next()); }

double nextDouble()

{

return Double.parseDouble(next());

}

String nextLine()

{

String str = "";

try {

if(st.hasMoreTokens()){

str = st.nextToken("\n");

}

else{

str = br.readLine();

}

}

catch (IOException e) {

e.printStackTrace();

}

return str;

}

}

7.Intermediate

/\* package codechef; // don't place package name! \*/

import java.util.\*;

import java.lang.\*;

import java.io.\*;

/\* Name of the class has to be "Main" only if the class is public. \*/

class Codechef

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

int n;

Scanner sc = new Scanner(System.in);

n = sc.nextInt();

while(n>0){

int length = 0;

length = sc.nextInt();

String s = sc.next();

System.out.println(function(s));

n--;

}

}

private static int function(String s){

int max = 1;

for(int i=0;i<s.length();i++){

//odd length;

int left = i;

int right = i;

while(left>=0 && right<s.length() && s.charAt(left)==s.charAt(right)){

left--;

right++;

}

max = Math.max(max,(right-1-left));

//even length

left=i;

right=i+1;

while(left>=0 && right<s.length() && s.charAt(left)==s.charAt(right)){

left--;

right++;

}

max = Math.max(max,(right-1-left));

}

return max;

}

}

8.Intermediate

import java.util.\*;

import java.lang.\*;

import java.io.\*;

class Codechef

{

public static void main (String[] args) throws java.lang.Exception

{

Scanner in=new Scanner(System.in);

int t=in.nextInt();

while(t-->0)

{

int n=in.nextInt();

String str=in.next();

char[] b=str.toCharArray();

int [] a=new int [26];

int min=Integer.MAX\_VALUE;

for(char x:b)

a[x-97]++;

for(int y:a)

{

if(y>0)

min=Math.min(min,y);

}

System.out.println(min);

}

}

}

9.Answer

import java.util.\*;

class Main {

public static void main(String []args){

LinkedList<Integer>list1=new LinkedList<Integer>();

Scanner sc = new Scanner(System.in);

int N = sc.nextInt();

int Q = sc.nextInt();

for(int i=0;i<N;i++){

list1.add(sc.nextInt());

}

while(Q-->0){

int a = sc.nextInt();

if(a==0){

int pos = sc.nextInt();

list1.remove(pos-1);

}else if(a==1){

int pos = sc.nextInt();

int value=sc.nextInt();

list1.add(pos-1,value);

}

for(int i=0;i<list1.size();i++){

System.out.print(list1.get(i)+" ");

}

System.out.println();

}

}

}

10.Answer

class Solution{

static int getLen(Node root){

Node curr = root;

int len = 0;

do{

curr = curr.next;

len++;

} while(curr!=root);

return len;

}

static int solve(Node root){

Node slow = root, fast = root;

while(fast!=null && fast.next!=null){

slow = slow.next;

fast = fast.next.next;

if(slow==fast){

return getLen(slow);

}

}

return -1;

}

}

HarshaVardhana M Ans :

Answer 1)

public class ConvertArrayListToArray {

public static void main(String args[]) {

// Creating and initializing ArrayList

ArrayList<String> fruits = new ArrayList<>();

fruits.add("Apple");

fruits.add("Banana");

fruits.add("Mango");

fruits.add("Pear");

// ArrayList to String array conversion

String[] str = new String[fruits.size()];

for(int i=0; i < fruits.size(); i++) {

str[i] = fruits.get(i);

}

// Print elements using for-each loop

for(String s : str) {

System.out.println(s);

}

}

}

Answer 2)

import java.util.\*;

class Main {

public static int ceilIndex(int[] nums, int target) {

int lo = 0; int hi = nums.length-1;

while(lo <= hi) {

int mid = lo + (hi-lo)/2;

if(nums[mid] == target) {

return mid;

} else if(nums[mid] < target) {

lo = mid + 1;

} else {

hi = mid - 1; } }

return lo; //ceil }

public static int search(int[] nums, int target) {

//insert position is actually the ceil of the element return ceilIndex(nums,target);

} public static void main(String args[])

{ // Your code goes here

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int[] arr = new int[n];

for(int i=0;i<n;i++) {

arr[i] = scn.nextInt();

}

int target = scn.nextInt();

System.out.println(search(arr,target));

}

}

Answer 3:

import java.util.\*;

class Main {

public static int countDigits(int n) {

if(n == 0)

return 1; //if a negative number is entered

if(n < 0)

n = -n;

int res = 0;

while(n != 0) {

n = n/10;

res++;

}

return res;

}

public static void main(String args[]) {

// Your code goes here

Scanner scn = new Scanner(System.in);

int n = scn.nextInt(); //input the number

System.out.println("The number of digits in " + n + " are: " + countDigits(n));

} }

Answer 4 :

import java.util.\*;

class Main {

public static int countDigitFreq(int n,int D) {

if(n == 0 && D == 0)

return 1; //number 0 has 1 frequency of 0

//if a negative number is entered

if(n < 0) n = -n;

int counter = 0;

while(n != 0) {

int digit = n % 10; //calculate the digit

if(digit == D)

counter++;

n = n/10;

}

return counter;

}

public static void main(String args[]) {

// Your code goes here

Scanner scn = new Scanner(System.in);

int n = scn.nextInt(); //input the number

int d = scn.nextInt(); //input the digit

int x = countDigitFreq(n,d);

System.out.println("The digit " + d + " occurs " + x + " times in " + n);

}

Answer 5 :

import java.util.\*;

class Main {

public static boolean isVowel(char ch) {

if(ch == 'a' || ch =='e' || ch =='i' || ch =='o' || ch =='u')

return true;

return false;

}

public static void main(String args[]) {

// Your code goes here

Scanner scn = new Scanner(System.in);

String str = scn.nextLine();

int vowelCount = 0;

int consCount = 0;

for(int i=0;i='a' && ch<='z' && isVowel(ch) == false)

consCount++;

}

System.out.println("Number of vowels are: " + vowelCount);

System.out.println("Number of consonants are: " + consCount);

System.out.println("Number of other characters are: " + (int)(str.length() - vowelCount -consCount));

}

}

Answer 6 :

import java.util.\*;

class Main {

public static void main(String args[]) {

// Your code goes here

Scanner scn = new Scanner(System.in);

String str = scn.nextLine();

HashSet unique = new HashSet<>();

for(int i=0;i<str.length();i++) {

char ch = str.charAt(i);

if(unique.contains(ch) == true) {

//this character has already occured

unique.remove(ch);

} else {

unique.add(ch);

}

}

if(unique.size() == 0) {

System.out.println("There are no unique characters");

}

for(Character ch : unique) {

System.out.print(ch + " ");

}

}

Answer 7 :

class Main { public static void main(String args[]) {

// Your code goes here

String s1 = "InterviewBit";

String s2 = s1;

System.out.println(s1 == s2);

//they are equal

s1 += "Scaler";

System.out.println(s1 == s2);

//not equal

}

}

Answer 8 :

import java.util.\*;

class Main {

public static void main(String args[]) {

// Your code goes here

Scanner scn = new Scanner(System.in);

int N = scn.nextInt();

int[][] mat = new int[N][N];

for(int i=0;i<N;i++) {

for(int j=0;j<N;j++) {

mat[i][j] = scn.nextInt();

}

}

//Display Row wise

for(int i=0;i<N;i++) {

System.out.print("Row " + i + " : ");

for(int j=0;j<N;j++) {

System.out.print(mat[i][j] + " ");

}

System.out.println("\t");

}

System.out.println();

//Display Col wise

for(int j=0;j<N;j++) {

System.out.print("Col " + j + " : ");

for(int i=0;i<N;i++) {

System.out.print(mat[i][j] + " ");

}

System.out.println("\t");

}

}

}

Answer 9 :

class Table {

public synchronized void display(int n) {

for (int i = 1; i <= 10; i++) {

System.out.println(n \* i);

}

}

}

class Thread1 extends Thread {

Table t;

public Thread1(Table t) {

this.t = t;

}

public void run() {

t.display(5);

}

}

class Thread2 extends Thread {

Table t;

public Thread2(Table t) {

this.t = t;

}

public void run() {

t.display(6);

}

} public class Main { public static void main(String[] args) {

Table table = new Table();

Thread1 th1 = new Thread1(table);

Thread2 th2 = new Thread2(table);

th1.start();

th2.start();

}

}

Mohan Ans:

1) import java.util.\*;

class Sports{

public static void main (String [] args){

String get\_name()

{

return "Generic Sports";

}

void get\_number\_of\_team\_members()

{

System.out.println("Each team has n players in "+get\_name());

}

}

class Soccer extends Sports

{

String get\_name()

{

return "Soccer Class";

}

//Complete the code

void get\_number\_of\_team\_members()

{

System.out.println("Each team has 11 players in "+get\_name());

}

}

public class Main

{

public static void main(String []args)

{

Sports C1=new Sports();

Soccer C2=new Soccer();

System.out.println(C1.get\_name());

C1.get\_number\_of\_team\_members();

System.out.println(C2.get\_name());

C2.get\_number\_of\_team\_members();

}

}

}

2) import java.util.\*;

class Student{}

class Rockstar{ }

class Hacker{}

public class InstanceOFTutorial{

static String count(ArrayList mylist){

int a = 0,b = 0,c = 0;

for(int i = 0; i < mylist.size(); i++){

Object element=mylist.get(i);

if(element instanceof Student)

a++;

if(element instanceof Rockstar)

b++;

if(element instanceof Hacker)

c++;

}

String ret = Integer.toString(a)+" "+ Integer.toString(b)+" "+ Integer.toString(c);

return ret;

}

public static void main(String []args){

ArrayList mylist = new ArrayList();

Scanner sc = new Scanner(System.in);

int t = sc.nextInt();

for(int i=0; i<t; i++){

String s=sc.next();

if(s.equals("Student"))mylist.add(new Student());

if(s.equals("Rockstar"))mylist.add(new Rockstar());

if(s.equals("Hacker"))mylist.add(new Hacker());

}

System.out.println(count(mylist));

}

}

3) PerformOperation isPrime() {

return num -> {

int count = 0;

for (int i = 1; i <= num; i++) {

if (num % i == 0) {

count++;

}

}

return count<=2?true:false;

};

}

PerformOperation isPalindrome() {

return num -> {

String str = Integer.toString(num);

String rev = "";

for (int i = str.length() - 1; i >= 0; i--) {

rev = rev + str.charAt(i);}

}

}

return str.equals(rev)?true:false;

};

}

4) public static void main(String [] args) {

int month, day, year, total=0;

Scanner sc=new Scanner(System.in);

System.out.print("Enter the year no. ");

year=sc.nextInt();

System.out.print("Enter the month no. ");

month=sc.nextInt();

System.out.print("Enter the date ");

day=sc.nextInt();

if(year%4==0)

{

if (month==1)

{

total=day;

}

if (month==2)

{

total=day+31;

}

if (month==3)

{

total=day+60;

}

if (month==4)

{

total=day+91;

}

if (month==5)

{

total=day+121;

}

if (month==6)

{

total=day+152;

}

if (month==7)

{

total=day+182;

}

if (month==8)

{

total=day+213;

}

if (month==9)

{

total=day+244;

}

if (month==10)

{

total=day+274;

}

if (month==11)

{

total=day+305;

}

if (month==12)

{

total=day+335;

}

System.out.print("Date given: "+day+"/"+month+"/"+year);

System.out.print("\nNo. of days: "+ total+" days");

}

if(year%4!=0)

{

if (month==1)

{

total=day;

}

if (month==2)

{

total=day+31;

}

if (month==3)

{

total=day+59;

}

if (month==4)

{

total=day+90;

}

if (month==5)

{

total=day+120;

}

if (month==6)

{

total=day+151;

}

if (month==7)

{

total=day+181;

}

if (month==8)

{

total=day+212;

}

if (month==9)

{

total=day+243;

}

if (month==10)

{

total=day+273;

}

if (month==11)

{

total=day+304;

}

if (month==12)

{

total=day+334;

}

System.out.print("Date given: "+day+"/"+month+"/"+year);

System.out.print("\nNo. of days: "+ total+" days");

}

}

5)

class Solution {

public static void main(String[] args) {

// Input

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

String[] s = new String[n];

for (int i = 0; i < n; i++) {

s[i] = sc.next();

}

sc.close();

// Write your code here

int len = s.length;

for (int i = 0; i < len - 1; ++i) {

for (int j = 0; j < len - i - 1; ++j) {

BigDecimal a = new BigDecimal(s[j + 1]);

BigDecimal b = new BigDecimal(s[j]);

if (a.compareTo(b) > 0) {

String temp = s[j + 1];

s[j + 1] = s[j];

s[j] = temp;

}

}

}

// Output

for (int i = 0; i < n; i++) {

System.out.println(s[i]);

}

}

}

6) public class Solution{

public static void main(String[] args){

Scanner in = new Scanner(System.in);

int testCases = Integer.parseInt(in.nextLine());

//System.out.println(testCases);

while(testCases>0 && in.hasNext()){

String line = in.nextLine();

Matcher matcher = Pattern.compile("<(.+)>(([^<>]+))</\\1>").matcher(line);

if (matcher.find() == false) {

System.out.println("None");

continue;

}

matcher.reset();

while (matcher.find() == true){

System.out.println(matcher.group(2));

}

}

in.close();

testCases--;

}

}

7) public static void main(String[] args){

Scanner in = new Scanner(System.in);

while(in.hasNext()){

String IP = in.next();

System.out.println(IP.matches(new MyRegex().pattern));

}

}

}

//Write your code here

class MyRegex{

public static String zeroTo255 = "(\\d{1,2}|(0|1)\\" + "d{2}|2[0-4]\\d|25[0-5])";

public static String pattern = zeroTo255 + "\\."+ zeroTo255 + "\\." + zeroTo255 + "\\." + zeroTo255;

Pattern p = Pattern.compile(pattern);

}

8) class Main

{

public static int LIS(int[] arr, int i, int n, int prev)

{

if (i == n) {

return 0;

}

int excl = LIS(arr, i + 1, n, prev);

int incl = 0;

if (arr[i] > prev) {

incl = 1 + LIS(arr, i + 1, n, arr[i]);

}

return Integer.max(incl, excl);

}

public static void main(String[] args)

{

int[] arr = { 15, 6, 5, 12, 1 };

System.out.print("The length of the LIS is "

+ LIS(arr, 0, arr.length, Integer.MIN\_VALUE));

}

# References

[Discussion Boards, Comments, Links, etc..]

Paste here.

Answers Swadhish

1 .

class Solution

{

//Function to find the maximum money the thief can get.

public int FindMaxSum(int arr[], int n)

{

//take a two variables i.e. prev1(just one index before) and prev2(just 2 index before)

int prev1=arr[0];

int prev2 =0;

//express the states for the loop

for(int i =1;i<n;i++){

int taken = arr[i]+prev2;

int notTaken = prev1;

int curr = Math.max(taken,notTaken);

//now for next index,curr becomes prev1 and previous prev2 becomes prev1

prev2 = prev1;

prev1 = curr;

}

return prev1;

}

}

References

[Discussion Boards, Comments, Links, etc..]

[https://practice.geeksforgeeks.org/problems/stickler-theif-1587115621/1?page=4&difficulty[]=0&difficulty[]=1&category[]=Arrays&category[]=Strings&category[]=Linked%20List&category[]=Java&sortBy=submissions](https://practice.geeksforgeeks.org/problems/stickler-theif-1587115621/1?page=4&difficulty%5b%5d=0&difficulty%5b%5d=1&category%5b%5d=Arrays&category%5b%5d=Strings&category%5b%5d=Linked%20List&category%5b%5d=Java&sortBy=submissions)

2. class Solution{

static Stack<Integer> deleteMid(int n, Stack<Integer> st)

{

LinkedList<Integer> ls = new LinkedList<Integer>(st);//Stack -> List

ls.remove(((n+1)/2)-1);//removes the mid element

st = new Stack<Integer>();//releases new memory for the stack

for(Integer i :ls)

st.push(i);//Creates a new Stack with all the elements that weren't removed

return st;//returns the stack

}

}

References

[Discussion Boards, Comments, Links, etc..]

[https://practice.geeksforgeeks.org/problems/java-collections-set-1-stack5707/1?page=1&difficulty[]=0&difficulty[]=1&category[]=Java&sortBy=submissions](https://practice.geeksforgeeks.org/problems/java-collections-set-1-stack5707/1?page=1&difficulty%5b%5d=0&difficulty%5b%5d=1&category%5b%5d=Java&sortBy=submissions)

3. class Solution {

int remove\_duplicate(int A[],int N){

//here we find non-duplicate consceutive elements

int sp =0;

for(int i=0;i<N-1;i++){

//if duplicates elements are came then we i++

if(A[i]==A[i+1])continue;

//we interesed in non-duplicate elements

if(A[i]!=A[i+1]){

A[sp]=A[i];

sp++;

}

}

A[sp++]=A[N-1];

return sp;

}

}

# References

[Discussion Boards, Comments, Links, etc..]

<https://practice.geeksforgeeks.org/problems/remove-duplicate-elements-from-sorted-array/1>

4. C++

#include <bits/stdc++.h>

using namespace std;

class Solution

{

public:

int no of open doors()

{

int open doors sqrt(N)

return doors open;

}

int main() {

int N = 10;

cout << countOpenDoors(N);

return 0;

}

# 

# References

[Discussion Boards, Comments, Links, etc..]

<https://practice.geeksforgeeks.org/problems/number-of-open-doors1552/1>

5. import java.io.\*;

import java.util.\*;

public class Main

{

// Fills f[] with first 300 fibonacci numbers

static void precomput(long f[])

{

/\* 0th and 1st number of the

series are 0 and 1\*/

f[0] = 0;

f[1] = 1;

/\* Add the previous 2 numbers in

the series and store last two

digits of result \*/

for (int i = 2; i < 300; i++)

f[i] = (f[i-1] + f[i-2]) % 100;

}

// Returns last two digits of Nth Fibonacci Number

static void findLastDigit(long f[], int n)

{

int s = (int)Math.abs(n%300);

if(f[s]<10)

System.out.println("0"+f[s]);

else

System.out.println(f[s]);

}

/\* Driver program to test above function \*/

public static void main (String args[])

{

// Precomputing units digit of first 300 Fibonacci numbers

long f[] = new long[300];

Arrays.fill(f,0);

precomput(f);

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

findLastDigit(f, n);

}

# References

[Discussion Boards, Comments, Links, etc..]

<https://practice.geeksforgeeks.org/problems/last-two-digit-fibonacci3353/1>

6. class Solution:

def getDayOfWeek(self, d, m, y):

weekday=calendar.weekday(y,m,d)

dict={'Monday':0, 'Tuesday':1,'Wednesday':2,'Thursday':3,'Friday':4,'Saturday':5,'Sunday':6}

key\_list = list(dict.keys())

val\_list = list(dict.values())

return(key\_list[val\_list.index(weekday)])

# References

[Discussion Boards, Comments, Links, etc..]

<https://practice.geeksforgeeks.org/problems/day-of-the-week1637/1>

7. class Hanoi {

public long toh(int N, int from, int to, int aux) {

// Your code here

long steps= (1<<N)-1;

if(N == 1){

System.out.println("move disk 1 from rod "+from+" to rod "+to);

return steps;

}

toh(N-1,from,aux,to);

System.out.println("move disk "+N+" from rod "+from+" to rod "+to);

toh(N-1,aux,to,from);

if( N == 1){

steps=1;

return steps;

} return steps;

}

}

References

[Discussion Boards, Comments, Links, etc..]

<https://practice.geeksforgeeks.org/problems/tower-of-hanoi-1587115621/1>

8. class Solution {

int binarysearch(int arr[], int n, int k) {

// code here

int li=0;

int hi=arr.length-1;

int mi=(li+hi)/2;

while(li<=hi){

if(arr[mi]==k){

return mi;}

else if(arr[mi]<k){

li=mi+1;}

else{

hi=mi-1;}

mi=(li+hi)/2;

}

return -1;

}

}

References

[Discussion Boards, Comments, Links, etc..]

<https://practice.geeksforgeeks.org/problems/binary-search-1587115620/1>

9 char[] arr1 = a.toCharArray();

char[] arr2 = b.toCharArray();

Arrays.sort(arr1);

Arrays.sort(arr2);

String str = new String();

String s1 = str.valueOf(arr1);

String s2 = str.valueOf(arr2);

return s1.equals(s2);

# References

[Discussion Boards, Comments, Links, etc..]

[https://practice.geeksforgeeks.org/problems/anagram-1587115620/1?page=2&difficulty[]=0&difficulty[]=1&category[]=Arrays&category[]=Strings&category[]=Linked%20List&category[]=Java&sortBy=submissions](https://practice.geeksforgeeks.org/problems/anagram-1587115620/1?page=2&difficulty%5b%5d=0&difficulty%5b%5d=1&category%5b%5d=Arrays&category%5b%5d=Strings&category%5b%5d=Linked%20List&category%5b%5d=Java&sortBy=submissions)

10. class Solution {

static int palindromepair(int N, String arr[]) {

// code here

for(int i=0;i<N;i++){

for(int j=i+1;j<N;j++){

String pal=arr[i]+arr[j];

String pal2=arr[j]+arr[i];

if(ispalindrome(pal) || ispalindrome(pal2)){

return 1;

}

}

}

return 0;

}

static boolean ispalindrome(String s){

for(int i=0;i<s.length();i++){

if(s.charAt(i)!=s.charAt(s.length()-i-1)){

return false;

}

}

return true;

}

}

References

[Discussion Boards, Comments, Links, etc..]

[https://practice.geeksforgeeks.org/problems/palindrome-pairs/1?page=1&difficulty[]=2&curated[]=1&sortBy=submissions](https://practice.geeksforgeeks.org/problems/palindrome-pairs/1?page=1&difficulty%5b%5d=2&curated%5b%5d=1&sortBy=submissions)