1. How Many X's?

Given an integer X within the range of 0 to 9, and given two positive integers as upper and lower bounds respectively, find the number of times X occurs as a digit in an integer within the range, excluding the bounds. Print the frequency of occurrence as output.

Note : Save the file / class as “countXInRange” with appropriate file extension.

Input:

The first line of input is an integer T, denoting the number of test cases. For each test case, there are two lines of input, first consisting of the integer X, whose occurrence has to be counted. Second, the lower and upper bound, L and U which are positive integers, on the same line separated by a single space, respectively.

Output:

For each test case, there is only one line of output, the count of the occurrence of X as a digit in the numbers lying between the lower and upper bound, excluding them.

Constraints:

1<=T<=100

0<=X<=9

0<L<U<=10^5

Example:

Input:

2

3

100 250

0

20 21

Output:

35

0

Explanation:

In the first test case, the occurrence of 3 in the numbers starting from 101 to 249 is counted and comes out to be 35.

Similarly, for all the other test cases, the occurrence of the given number X is printed as output.

Input:

3

2

10000 12345

9

899 1000

1

1100 1345

Output:

1120

120

398

2. Find the Minimum difference pair

Given an unsorted array, find the minimum difference between any pair in given array.

Note : Save the file / class as “minDifferencePair” with appropriate file extension.

Input:

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

The first line of each test case is N, the size of array. Second line of the test case is the Array.

Output:

Print the minimum difference between any two pairs.

Constraints:

1 <= T <= 30

1 < N <= 100

1 <= arr[i] <= 100000

Example:

Input:

2

5

2 4 5 7 9

10

87 32 99 75 56 43 21 10 68 49

Output:

1

6

3. Find the Maximum money

Given street of houses (a row of houses), each house having some amount of money kept inside; now there is a thief who is going to steal this money but he has a constraint/rule that he cannot steal/rob two adjacent houses. Find the maximum money he can rob.

Input:

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

The first line of each test case is N and money.

Output:

Print maximum money he can rob.

Note : Save the file / class as “maxMoney” with appropriate file extension.

Constraints:

1 ≤ T ≤ 100

1 ≤ money ≤ 100

1 ≤ N ≤ 1000

Example:

Input:

2

5 10

2 12

Output:

30

12

4. Rotate Array

Given an unsorted array arr[] of size N, rotate it by D elements (anti-clockwise).

Note : Save the file / class as “rotateArrayNTimes” with appropriate file extension.

Input:

The first line of the input contains T denoting the number of testcases. First line of eacg test case contains two space separated elements, N denoting the size of the array and an integer D denoting the number size of the rotation. Subsequent line will be the N space separated array elements.

Output:

For each testcase, in a new line, output the rotated array.

Constraints:

1 <= T <= 200

1 <= N <= 107

1 <= D <= N

0 <= arr[i] <= 105

Example:

Input:

2

5 2

1 2 3 4 5

10 3

2 4 6 8 10 12 14 16 18 20

Output:

3 4 5 1 2

8 10 12 14 16 18 20 2 4 6

Explanation :

Testcase 1: 1 2 3 4 5 when rotated by 2 elements, it becomes 3 4 5 1 2

5. Twice counter

Given an array of n words. Some words are repeated twice, we need count such words.

Note : Save the file / class as “wordTwiceCounter” with appropriate file extension.

Input:

The first line of input contains an integer T denoting the number of test cases. Then T test cases follow. Each test case contains an integer n denoting the number of words in the string. The next line contains n space separated words forming the string.

Output:

Print the count of the words which are repeated twice in the string.

Constraints:

1<=T<=105

1<=no of words<=105

1<=length of each word<=105

Example:

Input:

2

10

hate love peace love peace hate love peace love peace

8

Tom Jerry Thomas Tom Jerry Courage Tom Courage

Output:

1

2

6. Count total set/one bits

You are given a number N. Find the total count of set bits (number of 1s when the number is represented in binary) form for all numbers from 1 to N(both inclusive).

Note : Save the file / class as “countSetBits” with appropriate file extension.

Input:

The first line of input contains an integer T denoting the number of test cases. T testcases follow. The first line of each test case is N.

Output:

For each testcase, in a new line, print the total count of all bits.

Constraints:

1 ≤ T ≤ 100

1 ≤ N ≤ 103

Example:

Input:

2

4

17

Output:

5

35

Explanation:

Testcase1:

An easy way to look at it is to consider the number, n = 4:

0 0 0 = 0

0 0 1 = 1

0 1 0 = 1

0 1 1 = 2

1 0 0 = 1

Therefore , the total number of bits is 5.

7. Remove Characters in a String

Given two strings s1 and s2, remove those characters from first string which are present in second string. Both the strings are different and contain only lowercase characters.

Note : Save the file / class as “removeStringChars” with appropriate file extension.

Input:

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

The first line of each test case is s1,s1 is first string.

The second line of each test case contains s2,s2 is second string.

Output:

Print the modified string(s1). For each test case, print the output in a new line.

Constraints:

1 ≤ T ≤ 15

1 ≤ s2 < s1 ≤ 50

Example:

Input:

2

itvaccodingteam

caw

removeccharaterfrom

string

Output:

itvodingtem

emovecchaaefom

8. Generate Binary Numbers

Given a number N. The task is to generate and print all binary numbers with decimal values from 1 to N.

Note : Save the file / class as “genBinaryNos” with appropriate file extension.

Input:

The first line of input contains an integer T denoting the number of test cases. There will be a single line for each testcase which contains N.

Output:

Print all binary numbers with decimal values from 1 to N in a single line.

Constraints:

1 ≤ T ≤ 106

1 ≤ N ≤ 106

Example:

Input:

2

2

5

Output:

1 10

1 10 11 100 101

Explanation:

Testcase 1: Binary numbers from 1 to 2 are 1 and 10.

9. Keypad typing

You are given a string S of alphabet characters and the task is to find its matching decimal representation as on a mobile phone's numeric keypad. Output the decimal representation corresponding to the string. For ex: if you are given “amazon” then its corresponding decimal representation will be 262966.

Note : Save the file / class as “stringKeyMapping” with appropriate file extension.

Input:

The first line of input contains an integer T denoting the number of test cases. Then T test cases follow. Each test case consists of a single line containing a string.

Output:

For each test case, print in a new line, the corresponding decimal representation of the given string.

Reference



Constraints:

1 ≤ T ≤ 100

1 ≤ length of String ≤ 100

Example:

Input

2

itvaccoding

vacquiz

Output

48822263464

8227849

10. Greater on right side

You are given an array A of size N. Replace every element with the next greatest element (greatest element on its right side) in the array. Also, if there is no element next to the last element, replace it with -1.

Note : Save the file / class as “greatOnRight” with appropriate file extension.

Input:

The first line of input contains an integer T denoting the number of test cases. T testcases follow. Each test-case contains two lines of input. The first line is N, the size of tha array. The second line contains N space separated integers.

Output:

For each test-case, print the modified array.

Constraints:

1 <= T <= 50

1 <= N <= 100

1 <= Ai <= 1000

Example:

Input:

2

6

16 17 4 3 5 2

4

2 3 1 9

Output:

17 5 5 5 2 -1

9 9 9 -1

Explanation:

Testcase1: For 16 the greatest element on its right is 17. For 17 it's 5. For 4 it's 5. For 3 it's 5. For 5 it's 2. For 2 it's -1(no element to its right). So the answer is 17 5 5 5 2 -1

11. Addition of sub-matrix

Given a matrix C of size N x M. You are given position of sub-matrix as X1, Y1 and X2, Y2 inside the matrix. Find the sum of all elements inside that sub-matrix.

Note : Save the file / class as “addSubMatrix” with appropriate file extension.

Input:

The first line of input contains an integer T denoting the number of test cases. The first line of each test case is n and m,n is the number of rows and m is the number of columns. The second line of each test case contains C[N][M]. The third line contains four value of X1, Y1, X2, Y2. X1, Y1 is the top left cell and X2, Y2 is the bottom right cell.

Output:

Print the sum of all elements inside that sub-matrix.

Constraints:

1 ≤ T ≤ 15

1 ≤ N, M ≤ 103

1 ≤ C[N][M] ≤ 106

1 <= X1, Y1, X2, Y2 <= M

Example:

Input:

2

5 6

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

3 4 4 5

3 3

9 8 7 4 2 1 6 5 3

1 2 3 3

Output:

78

26

Explanation:

Testcase 2: Sum from cell starting at position (1, 2) (1-based indexing) and ending at (3, 3) is 26.

12. Geek and Coffee Shop

Geek loves to drink cold coffee after coding for long hours. One fine day a Geek went to his favorite coffee shop in the town and asked for a cup of cold coffee. The shopkeeper told him that he is their lucky customer and had an offer for the Geek. The offer was that for an amount of N they will fill Geek's cup with N units of cold coffee and after consuming initial N units of coffee the shopkeeper will again refill his cup with half the amount of coffee that Geek consumed in previous refill, and will keep on refilling his cup by half of the previous cycle till the amount to refill becomes nil i.e. 0 (Assume Geek can consume infinite amount of coffee and shop also has infinite amount coffee). Now Geek is curious to know that how many units of coffee will Geek get for Mth refill. Being Geek's friend help him out with his problem.

Note : Save the file / class as “geeksCoffee” with appropriate file extension.

Input:

The first line of the input contains an integer T, denoting the number of test cases. The T test case follow. The only line of each test case contains two space separated integers N and M respectively.

Output:

For each test case output a single integer on a new line denoting the required answer.

Constraints:

1<=T<=104

1<=N<=109

1<=M<=103

Example:

Input:

2

100 4

10 3

Output:

12

2

Explanation:

TestCase 1:

For the 4th refill geek will get 12 units of the coffee.

1st Fill: Geek will get 100 units of Coffee

2nd Fill: 100/2 = 50 units

3rd Fill: 50/2 = 25 units

4th Fill: 25/2 = 12 units

13. Maximum product of two numbers

Given an array with all elements greater than or equal to zero. Return the maximum product of two numbers possible.

Note : Save the file / class as “maxProductInArray” with appropriate file extension.

Input:

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

The first line of each test case is N, size of array. The second line of each test case contains array elements.

Output:

Print the maximum product of two numbers possible.

Constraints:

1 ≤ T ≤ 100

2 ≤ N ≤ 107

0 ≤ A[i] ≤ 104

Example:

Input:

2

5

1 100 42 4 23

3

20 30 40

Output:

4200

1200

Explanation:

Testcase 1: Two maximum numbers are 100 and 42 and their product is 4200.

14. Decode the pattern

Given a pattern as below and an integer n your task is to decode it and print nth row of it. The pattern follows as :

11

121

1331

14641

Note : Save the file / class as “oneOnePattern” with appropriate file extension.

Input:

The first line of input is the number of test cases . Then T test cases follow . The first line of each test case is an integer N.

Output:

For each test case print the required nth row of the pattern.

Constraints:

1<=T<=20

1<=N<=20

Example:

Input:

2

2

4

Output:

11

121

11

121

1331

14641

15. Number following a pattern

Given a pattern containing only I's and D's. I for increasing and D for decreasing.

Devise an algorithm to print the minimum number following that pattern.

Digits from 1-9 and digits can't repeat.

Input:

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

The first line of each test case is a string, which contains only I's and D's in capital letter.

Output:

Print the minimum number following that pattern.

Constraints:

1 ≤ T ≤ 100

1 ≤ Length of String ≤ 8

Example:

Input

5

D

I

DD

IIDDD

DDIDDIID

Output

21

12

321

126543

321654798

16. Convert array into Zig-Zag fashion

Given an array A (distinct elements) of size N. Rearrange the elements of array in zig-zag fashion. The converted array should be in form a < b > c < d > e < f. The relative order of elements is same in the output i.e you have to iterate on the original array only.

Input:

The first line of input contains an integer T denoting the number of test cases. T testcases follow. Each testcase contains two lines of input. The first line contains a single integer N denoting the size of array.

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

Output:

For each testcase, print the array in Zig-Zag fashion.

Constraints:

1 <= T <= 100

1 <= N <= 100

0 <= Ai <= 10000

Example:

Input:

2

7

4 3 7 8 6 2 1

4

1 4 3 2

Output:

3 7 4 8 2 6 1

1 4 2 3

17. Implement strstr manually

Your task is to implement the function strstr. The function takes two strings as arguments (s,x) and locates the occurrence of the string x in the string s. The function returns and integer denoting the first occurrence of the string x in s.

Note : Save the file / class as “manualSubString” with appropriate file extension.

Input Format:

The first line of input contains an integer T denoting the no of test cases . Then T test cases follow. Each test case has 2 lines of input, first list the search string s and second line the occurrence of string x

Output Format:

For each test case, in a new line, output will be an integer denoting the first occurrence of the x in the string s. Return -1 if no match found.  
NOTE: If there is a match on the first element answer should be 0 and not 1 given that array index starts at 0

Constraints:

1 <= T <= 100

1<= |s|,|x| <= 1000

Example:

Input

2

ItVACForSuccess

Fr

ITVACForsuccess

For

Output

-1

5

Explanation:

Testcase 1: Fr is not present in the string ItVACForSuccess as substring.

Testcase 2: For is present as substring in ITVACForsuccess from index 5 given that array index starts at 0.

18. Binary String

Given a binary string S. The task is to count the number of substrings that start and end with 1. For example, if the input string is “00100101”, then there are three substrings “1001”, “100101” and “101”.

Note : Save the file / class as “binaryStartAndEnd” with appropriate file extension.

Input:

The first line of input contains an integer T denoting the number of test cases. Each test case consist of an integer 'n' denoting the string length and next line is followed by a binary string.

Output:

For each testcase, in a new line, print the number of substring starting and ending with 1 in a separate line.

Constraints:

1 ≤ T ≤ 40

1 ≤ |S| ≤ 1000

Example:

Input:

2

4

1111

5

01101

Output:

6

3

Example:

Testcase 1: There are 6 substrings from the given string. They are 11, 11, 11, 111, 111, 1111.

Testcase 2: There 3 substrings from the given string. They are 11, 101, 1101.

19. Sum of Lengths of Non-Overlapping SubArrays

Given an array of N elements, you are required to find the maximum sum of lengths of all non-overlapping subarrays with K as the maximum element in the subarray.

.

Input:

First line of the input contains an integer T, denoting the number of the total test cases. Then T test case follows. First line of the test case contains an integer N, denoting the number of elements in the array. Then next line contains N space separated integers denoting the elements of the array. The last line of each test case contains an integer K.

Output:

For each test case ouptut a single line denoting the sum of the length of all such subarrays.

Constraints:

1 <= T <= 100

1 <= N <= 105

1 <= A[] <= 105

Example:

Input:

3

9

2 1 4 9 2 3 8 3 4

4

7

1 2 3 2 3 4 1

4

10

4 5 7 1 2 9 8 4 3 1

4

Output:

5

7

4

Explanation:

Test Case 1:

{2, 1, 4} => Length = 3

{3, 4} => Length = 2

So, 3 + 2 = 5 is the answer

Test Case 2: {1, 2, 3, 2, 3, 4, 1} => Length = 7

Test Case 3:

{4} => Length = 1

{4, 3, 1} => Length = 3

So, 1 + 3 = 4 is the answer.

20. Reverse each word in a given string

Given a String of length N reverse each word in it. Words are separated by spaces. Dot character should not be reversed and should retain its position.

Note : Save the file / class as “reverseEachWord” with appropriate file extension.

Input:

The first line contains T denoting the number of testcases. Then follows description of testcases. Each case contains a string containing dots, spaces and characters.

Output:

For each test case, output a String in single line containing the reversed words of the given String.

Constraints:

1<=T<=10

1<=Length of String<=2000

Example:

Input:

2

i like this program very much

hello

Output:

i ekil siht margorp yrev hcum

olleh

21. Given a String of length N capitalize the first letter of each word in the sentence

Input:

The first line contains T denoting the number of testcases. Then follows description of testcases. Each case contains a string containing dots and characters.

Note : Save the file / class as “capitalizeFirstLetter” with appropriate file extension.

Output:

For each test case, output a String in single line containing the words of the given String with first letter as upper case (capitals)

Constraints:

1<=T<=10

1<=Length of String<=2000

Example:

Input:

2

i like this. program very much

….

Output:

I Like This. Program Very Much

…

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Sorting Elements of an Array by Frequency

Given an array A[] of integers, sort the array according to frequency of elements. That is elements that have higher frequency come first. If frequencies of two elements are same, then smaller number comes first.

Input:

The first line of input contains an integer T denoting the number of test cases. The description of T test cases follows. The first line of each test case contains a single integer N denoting the size of array. The second line contains N space-separated integers A1, A2, ..., AN denoting the elements of the array.

Output:

For each testcase, in a new line, print each sorted array in a seperate line. For each array its numbers should be seperated by space.

Constraints:

1 ≤ T ≤ 70

30 ≤ N ≤ 130

1 ≤ Ai ≤ 60

Example:

Input:

2

5

5 5 4 6 4

5

9 9 9 2 5

Output:

4 4 5 5 6

9 9 9 2 5

Explanation:

Testcase1: The highest frequency here is 2. Both 5 and 4 have that frequency. Now since the frequencies are same then smaller element comes first. So 4 4 comes first then comes 5 5. Finally comes 6.

The output is 4 4 5 5 6.

Testcase2: The highest frequency here is 3. The element 9 has the highest frequency. So 9 9 9 comes first. Now both 2 and 5 have same frequency. So we print smaller element first.

The output is 9 9 9 2 5.

2. Extract the Number from the String

Bastin once had trouble finding the numbers in a string. The numbers are distributed in a string across various test cases.

There are various numbers in each test case you need to find the number in each test case. Each test case has various numbers in sequence. You need to find only those numbers which do not contain 9. For eg, if the string contains "hello this is alpha 5051 and 9475".You will extract 5051 and not 9475. You need only those numbers which are consecutive and you need to help him find the numbers.

Note: Use long long for storing the numbers from the string.

Input:

The first line consists of T test cases and next T lines contain a string.

Output:

For each string output the number stored in that string if various numbers are there print the largest one. If a string has no numbers print -1.

Constraints:

1<=T<=100

1<=|S|<=10000

Example:

Input:

1

This is alpha 5057 and 97

Output:

5057

3. Valid Parenthesis String

Given a string containing only three types of characters: '(', ')' and '\*', write a function to check whether this string is valid. We define the validity of a string by these rules:

Any left parenthesis '(' must have a corresponding right parenthesis ')'.

Any right parenthesis ')' must have a corresponding left parenthesis '('.

Left parenthesis '(' must go before the corresponding right parenthesis ')'.

'\*' could be treated as a single right parenthesis ')' or a single left parenthesis '(' or an empty string.

An empty string is also valid.

Example 1:

Input: "()"

Output: True

Example 2:

Input: "(\*)"

Output: True

Example 3:

Input: "(\*))"

Output: True

Note:

The string size will be in the range [1, 100].

4. Time Conversion

Given a time in 12[-hour AM/PM format](https://en.wikipedia.org/wiki/12-hour_clock), convert it to military (24-hour) time.

Note: Midnight is 12:00:00AM on a 12-hour clock, and 00:00:00 on a 24-hour clock. Noon is 12:00:00PM on a 12-hour clock, and 12:00:00 on a 24-hour clock.

Input Format

A single string s containing a time in 12-hour clock format (i.e.: hh:mm:ssAM or hh:mm:ssPM ), where 01<= hh <= 12 and 00 <= mm, ss <= 59.

Constraints

All input times are valid

Output Format

Convert and print the given time in 24-hour format, where 00 <= hh <= 23.

Sample Input 0

07:05:45PM

Sample Output 0

19:05:45

5. You have been asked to help study the population of birds migrating across the continent. Each type of bird you are interested in will be identified by an integer value. Each time a particular kind of bird is spotted, its id number will be added to your array of sightings. You would like to be able to find out which type of bird is most common given a list of sightings. Your task is to print the type number of that bird and if two or more types of birds are equally common, choose the type with the smallest ID number.

For example, assume your bird sightings are of types arr=[1,1,2,2,3]. There are two each of types 1 and 2 , and one sighting of type 3. Pick the lower of the two types seen twice: type 1.

**Function Description**

Complete the *migratoryBirds* function in the editor below. It should return the lowest type number of the most frequently sighted bird.

migratoryBirds has the following parameter(s):

* *arr*: an array of integers representing types of birds sighted

**Input Format**

The first line contains an integer denoting n, the number of birds sighted and reported in the array arr.   
The second line describes arr as n space-separated integers representing the type numbers of each bird sighted.

**Constraints**

* 5 <= n <= 2\*10^5
* It is guaranteed that each type is 1, 2, 3, 4, or 5.

**Output Format**

Print the type number of the most common bird; if two or more types of birds are equally common, choose the type with the smallest ID number.

**Sample Input 0**

6

1 4 4 4 5 3

**Sample Output 0**

4

**Explanation 0**

The different types of birds occur in the following frequencies:

* Type 1: 1 bird
* Type 2: 0 birds
* Type 3: 1 bird
* Type 4: 3 birds
* Type 5: 1 bird

The type number that occurs at the highest frequency is type , so we print  as our answer.

**Sample Input 1**

11

1 2 3 4 5 4 3 2 1 3 4

**Sample Output 1**

3

**Explanation 1**

The different types of birds occur in the following frequencies:

* Type 1: 2 birds
* Type 2: 2 birds
* Type 3: 3 birds
* Type 4: 3 birds
* Type 5: 1 bird

Two types have a frequency of 3, and the lower of those is type 4.