**Question 1:-**

Ramu has N dishes of different types arranged in a row: A1,A2,…,AN where Ai denotes the type of the ith dish. He wants to choose as many dishes as possible from the given list but while satisfying two conditions:

He can choose only one type of dish.

No two chosen dishes should be adjacent to each other.

Ramu wants to know which type of dish he should choose from, so that he can pick the maximum number of dishes.

**Example :**

Given N= 9 and A= [1,2,2,1,2,1,1,1,1]

For type 1, Ramu can choose at most four dishes. One of the ways to choose four dishes of type 1 is A1,A4, A7 and A9.

For type 2, Ramu can choose at most two dishes. One way is to choose A3 and A5.

So in this case, Ramu should go for type 1, in which he can pick more dishes.

**INPUT FORMAT**

The first line contains T, the number of test cases. Then the test cases follow.

For each test case, the first line contains a single integer N.

The second line contains N integers A1,A2,…,AN.

**OUTPUT FORMAT**

For each test case, print a single line containing one integer ― the type of the dish that Ramu should choose from. If there are multiple answers, print the smallest one.

**CONSTRAINTS :**

1 <= T <= 10^3

1 <= N <= 10^3

1 <= Ai <= 10^3

**Sample Input :**

3

5

1 2 2 1 2

6

1 1 1 1 1 1

8

1 2 2 2 3 4 2 1

**Sample Output :**

1

1

2

**Question 2:-**

There are three piles of stones. The first pile contains a stones, the second pile contains b stones and the third pile contains c stones. You must choose one of the piles and split the stones from it to the other two piles; specifically, if the chosen pile initially contained s stones, you should choose an integer k (0≤k≤s), move k stones from the chosen pile onto one of the remaining two piles and s−k stones onto the other remaining pile. Determine if it is possible for the two remaining piles (in any order) to contain x stones and y stones respectively after performing

this action.

**INPUT FORMAT :**

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

The first and only line of each test case contains five space-separated integers

a,b,c, x and y.

**OUTPUT FORMAT :**

For each test case, print a single line containing the string “YES” if it is possible to obtain piles of the given sizes or “NO” if it is impossible.

**CONSTRAINTS :**

1 <= T <= 100

1 <= a,b,c,x,y <= 10^9

**SAMPLE INPUT :**

4

1 2 3 2 4

3 2 5 6 5

2 4 2 6 2

6 5 2 12 1

**SAMPLE OUTPUT :**

YES

NO

YES

NO

Test case 1: You can take the two stones on the second pile, put one of them on the first pile and the other one on the third pile.

Test case 2: You do not have enough stones.

Test case 3: You can choose the first pile and put all stones from it on the second pile.

**Question 3:-**

Altaf has recently learned about number bases and is becoming fascinated.

Altaf learned that for bases greater than ten, new digit symbols need to be introduced, and that the convention is to use the first few letters of the English alphabet. For example, in base 16, the digits are 0123456789ABCDEF. Altaf thought that this is unsustainable; the English alphabet only has 26 letters, so this scheme can only work up to base 36. But this is no problem for Altaf, because Altaf is very creative and can just invent new digit symbols when she needs them. (Altaf is very creative.)

Altaf also noticed that in base two, all positive integers start with the digit 1! However, this is the only base where this is true. So naturally, Altaf wonders: Given some integer N, how many bases b are there such that the base-b representation of N starts with a 1?

**INPUT FORMAT :**

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

Each test case consists of one line containing a single integer N (in base ten).

**OUTPUT FORMAT :**

For each test case, output a single line containing the number of bases b, or INFINITY if there are an infinite number of them.

**CONSTRAINTS :**

1 <= T <= 10^5

0 <= N < 10^12

**SAMPLE INPUT :**

4

6

9

11

24

**SAMPLE OUTPUT :**

4

7

8

14

**Question 4 :-**

**Paper Sheets**

Sam is given a rectangular paper having dimensions h\*w, where h is the height and w is the width. Sam wants to fold the paper so its dimensions are h1\*w1 in the minimum number of moves. The paper can only be folded parallel to its edges and after folding, the dimensions should be integers.

For example, given h=8 and w=4, fold the paper until it is h1, w1 = 6, 1. First fold along the long edge 6 units from a side, resulting in a paper that is 6\*4. Next, fold along the width 2 units from the 4 unit edge to have 6\*2. Fold along the center of the 2 unit edge to achieve a 6\*1 page in three folds.

minMoves has following parameters :

h : an integer that denotes the initial height

w : an integer that denotes the initial width

h1 : an integer that denotes the final height

w1 : an integer that denotes the final width

**Constraints**

1 <= h, w, h1, w1 <= 10^15

h1 <= h

w1 <= w

**Sample Input 0**

2

3

2

2

**Sample Output**

1

**Question 5 :-**

**Break the Waffle**

Charlie wants to divide a big piece of waffle which is made up of m\*n square pieces. Each piece is of size 1\*1. The shape of waffle is a rectangle of dimension **m\*n**. However, Charlie can break the waffle either horizontally or vertically, along the lines such that the square pieces are not destroyed.

Each break along a line has a certain cost associated with it. The cost is only dependent on the line along which the waffle is being broken, and not on its length.

The total cost is obtained by summing up the costs of all the breaks.

Given the cost associated with each line, Charlie wants to know the minimum cost of breaking the waffle into single square pieces.

**Input**

First line contains two integers **m** and **n** denoting the number of rows and columns respectively.

This is followed by m-1, each line contains an integer denoting the cost of breaking a waffle along a horizontal line.

This is followed by n-1, each line contains an integer denoting the cost of breaking a waffle along a vertical line.

**Output**

Output should be a single line integer denoting the minimum cost to break the waffle into single square pieces.

**Constraints**

1<=n<=10^5

1<=m<=10^5

**Sample Input**

2 2

3

4

**Sample Output**

10

**Explanation**

Break the waffer along the column where the cost = 4, then break the two pieces along the row where the cost = 3\*2 = 6. Otherwise, the cost would have been 3+(4\*2) = 11.

Question 6 :-

**Dangerous Script**

If you have ever looked at an assembly language dump of an executable file, you know that commands come in the form of hexadecimal digits that are grouped by the processor into instructions. It is important that parsing can be done correctly or code will not be executed as expected. Wrong Parsing is the basis for Return Oriented Programming.

You have developed a program in a new scripting language. Of course, it requires accurate parsing in order to perform as expeced, and it is very cryptic. You want to determine how many valid commands can be made out of your lines of script. To do this, you count all of the substrings that make up a valid command. Each of the valid commands will be in the following format :

1. First letter is a lowercase English letter

2. Next, it contains a sequence of zero or more of the following characters : lowercase English letters, digits, and colons.

3. Next, it contains a forward slash ‘/’.

4. Next, it contains a sequence of one or more of the following characters : lowercase English letters, digits.

5. Next, it contains a backward slash ‘\’.

6. Next, it contains a sequence of one or more lowercase English letter.

Given some string s, we define the following :

s[i…j] is a substring containing of all the characters in the inclusive range between i and j.

For example, your command line is abc:/b1c\xy. Valid command There are six valid commands that may be parsed from that one command string.

**Sample Input 0**

6

w\\//a/b

w\\//a\b

w\\/a\b

w:://a\b

w::/a\b

w:/a\bc::/12\xyz

**Sample Output**

0

0

0

0

1

8

Question 7 :-

**Uniformity**

You are given a string that is formed from only three characters ‘a’, ‘b’, ‘c’. You are allowed to change atmost ‘k’ characters in the given string while attempting to optimize the uniformity index.

Note : The uniformity index of a string is defined by the maximum length of the substring that contains same character in it.

**Input**

The first line of input contains two integers **n** (the size of string) and **k**. The next line contains a string of length **n**.

**Output**

A single integer denoting the maximum uniformity index that can be achieved.

Question 8 :-

**Choose a Flask**

An olive oil seller needs to measure oi; for customers using only one type of flask. There are many flasks available, each with marking at various levels. Each customer must receive a flask filled to a mark that is atleast equal to the amount ordered. Given a list of customer requirements and a list of flasks with their measurements, determine the single type of flask that will result in the minimal loss to the merchant. Loss is the sum of marking – requirement for each order. Return the zero-based index of the flask type chosen. If there are multiple answers, return the minimum index. If no flask will satisfy the constraints, return -1.

For example, there are n=4 orders for requirements = [4, 6, 6, 7] units of oil. There are m=3 types of flask available with markings = [“3 5 7”, “6 8 9”, “3 5 6”]. These markings are given as 2D array with total\_marks rows and 2 columns, first is the index of flask and second the mark. To fill the orders using the flask at markings[0] = [3,5,7], the loss is calculated as marking requirement for each order so, 5-4=1, 7-6=1 and 7-7=0. The total loss then is 1+1+1+0=3. Choosing the flask at markings[1], the loss is 6-4=2, 6-6=0, 8-7=1 -> 2+0+0+1 = 3. The third flask has a maximum mark at 6 which is smaller than the largest order, so it cannot be used. In this case, flask type 0 is chosen.

**Note** ! The markings 2d array will be given in order of the flasks, i.e the markings for the 0-index flask will be followed by markings of 1-index flask and so on. For each flask, the given markings will also be in the sorted order.

**Constraints**

1 <= n <= 100

-1 <= mat[i][j] <= 1

**Sample Input 0**

2

4

6

2

5

2

0 5

0 7

0 10

1 4

1 10

**Sample Output 0 :**

0

**Constraints**

1 <= n <= 10^6

0 <= k <= n

String contains only ‘a’, ‘b’, ‘c’.

**Sample Input 0**

6 3

abaccc

**Sample Output 0**

6

**Explanation**

First 3 letters can be changed to ‘c’ and we can get the string ‘cccccc’

Question 9 :-

**Weighted Strings**

In Hackerland every character has a weight. The weight of an English uppercase alphabet A-Z is given below :

A = 1

B = 2\*A + A

C = 3\*B + B

D = 4\*C + C

….

Z = 26\*Y + Y

The weight made up of these characters is the summation of weights of each character. Given a total string weight, determine shortest string of given weight. If there is more than one solution, return the lexicographically smallest of them. For example, given weight = 25, and the weights of the first few characters of the alphabets are A=1, B=3, C=12, D=60 it is certain that no letter larger than C is required. Some of the strings with a total weight equal to the target are ABBBBC, ACC, AAAAAAABBBBBB. The shortest of these is ACC. While any permutation of these characters will have same weight, this is the lexicographically smallest of them.

**Example**

**Input**

20

**Output**

AABBC

Question 10 :-

**Class Students**

Dia, Sam, and Robert are the three students of a same class. You know their marks in ‘N’ subjects. Your job is to find their ranks according to their score.

**Input**

N, integer denoting number of subjects.  
3 array of integers, denoting marks of Dia, Sam and Robert respectively in N subjects.

**Output**

Ranks

**Example**

**Input**

3  
23 34 23  
32 53 23  
55 22 67

**Output**

3 2 1

Question 11:-

**Distinct Substrings**

Given a string, count all distinct substrings of the given string.

**Example**

**Input**

abcd

**Output**

10

All Elements are Distinct

**Input**

aaa

**Output**

6

Question 12:-

**Tree Edges**

Given a tree with N nodes we are required to seperate a connected component with exactly k nodes. You are given queries specifying this k. We need to find the minimum edges to be removed for each query.

**Input**

First line specifies N.  
Next N-1 lines specify edges.  
Next line shows Q(number of queries).  
Subsequent Q lines contain k for each query.

**Constraint**

N <= 3000  
Q <= 3000  
K <= N

**Example**

**Input**

5  
1 2  
1 3  
1 4  
1 5  
3  
1  
2  
4

**Output**

1  
3  
1

Question 13 :-

**Longest SubArray**

Consider an array A. Your job is to find longest subarray in which elements greater than x are more than elements not greater than x

**Input**

1. size of array , x  
2. Array elements

**Example**

**Input**

5 5  
4 5 7 8 3

**Output**

3

**Explanation**

Subarray formed : [5,7,8]

Question 14 :-

**m\*n board**

Find the total number of ways a m×n board can be painted using 3 colors while making sure no cells of the same row or the same column have entirely the same color. The answer must be computed modulo 10^9+7.

**Input**

Two integers – m, n respectively

**Example**

**Input**

3 2

**Output**

174

Question 15 :-

**String Rotation**

A rotation on a string is defined as removing first element and concatenating it at the end.

Given N and an array of N strings.  
Your job is to predict the minimum no. of rotations on the strings so as to make all the strings equal.  
If this is not possible return -1

**Example**

**Input**

4  
11234  
34112  
41123  
11234

**Output**

3

Question 16 :-

**Even-Odd Problem**

Given L , R ( 1<=L,R <= 1000000000000000000) . Count the numbers in [L,R] such that each number satisfies two properties :

1. The total count of odd digits are even

2. The total count of even digits are odd

Note – Do not consider zero in MSB position

**Testcase**

**Input**

1 9

**Output**

4

**Explanation**

The numbers that satisfy both the properties are 2,4,6,8 . In each of these numbers , the count of odd digits are even (i.e 0 ) and count of even digits are odd (i.e 1 ).

Question 17 :-

**Special Prime**

Given an integer N  
N<=10^50  
Output the no. of pairs (x,y) such that

0<=x<=n  
0<=y<=n  
F(x) +F(y) = Prime number

F(x) = sum of digits of x

**Note** : (x,y) and (y,x) are to be treated as same pair

**Example**

**Input**

3

**Output**

5

**Explanation**

5 pairs (0,2), (1,1), (0,3), (2,3), (1,2) give prime no.s

Question 18 :-

**Particle Simulation**

Given an integer(n) denoting the no. of particles initially and array of sizes of these particles.

These particles can go into any number of simulations (possibly none).

In one simulation, two particles combine to give another particle with size as the difference between the size of them (possibly 0).

Find the smallest particle that can be formed.

**Constraints**

n<=1000

size<=10^9

**Example 1**

**Input**

3

30 10 8

**Output**

2

**Explanation**

10 – 8 is the smallest we can achieve.

**Example 2**

**Input**

4

1 2 4 8

**Output**

1

**Explanation**

We cannot make another 1 so as to get 0 so smallest without any simulation is 1

**Example 3**

**Input**

5

30 27 26 10 6

**Output**

0

Question 19 :-

**Game of Pile**

Sam and Alex are playing a new game where there are number of piles, each with any number of stones in it. Players take turns removing stones from any one pile.

The number removed has to be either an integer multiple of a given number, k, where k > 0.  
If there are fewer than k stones in a pile, any number can be removed.  
Determine who wins the game.

Sam always starts, they both play optimally, and the last player to remove a stone wins. If Sam wins, return ” Sam wins the game of n pile(s). “, where n is the number of piles in the input. If Alex wins, return ” Alex wins the game of n pile(s). “.

For example, a game starts with n = 3 piles of stones that contain piles = [3, 5, 7] stones, and a constant k = 2. Sam will go first and remove 1 \* k = 1 \* 2 stones from the third pile leaving 7 – 2 = 5 stones in that pile.

Player Removes Result Start [3, 5, 7] Sam 2 [3, 5, 5] Alex 2 [3, 3, 5] Sam 2 [3, 3, 3] Alex 2 [1, 3, 3] Sam 2 [1, 1, 3] Alex 2 [1, 1, 1] Sam 1 [0, 1, 1] Alex 1 [0, 0, 1] Sam 1 [0, 0, 0]

In this case, Sam wins so ” Sam wins the game of 3 pile(s). ” is returned.

**Function Description**

Complete the function **gameOfPiles** in the editor below. The function must return a string that denotes the result of the game.

**gameOfPiles** has the following parameters:

1. piles[piles[0], piles[1],…piles[n-1]] : an array of integers, each piles[i] (where 0 ≤ i < n ) represents the number of stones in the i-th pile.  
2. k: an integer

**Constraints**

1 ≤ n ≤ 105  
1 ≤ piles[i] ≤ 1000  
1 ≤ k ≤ 1000  
Input Format For Custom Testing Sample Case 0

**Input For Custom Testing**

2 2 2 1

**Output**

Alex wins the game of 2 pile(s).

**Explanation**

There are multiple optimal scenarios for this game. Here is a scenario as an example where k = 1 .

Sam first removes 1 stone from the first pile, the configuration becomes [1, 2] .

Alex then removes 1 stone from the second pile, the configuration becomes [1, 1] .

Sam then removes 1 stone from the first pile, the configuration becomes [0, 1] .

At last, Alex removes 1 stone from the second pile, the configuration becomes [0, 0] .  
Alex makes the last move so Alex wins.