58/62 BD SN Toph β

Problem A

Ladiohead

Input: Standard Input
Output: Standard Output

You are a Radiohead fangirl and want to have a band of your own named **Ladiohead**. You have convinced three of your friends to join this band. Now you're charged with arranging the guitars and speakers required for functioning properly.

As large speakers are somewhat costly, you have decided to build your own for which you need magnets. Magnets can be found at the local hardware store and their cost is proportional to their size (i.e., their radius, as speaker magnets are circular). Magnets are disk-shaped with a fixed height of **1** unit and a radius **R** unit. As always, you bargained heavily with the shopkeeper and came to a term that you will only pay to buy a bag of size **r**, and won't pay individually for magnets. An **r**-sized bag can carry magnets that have a radius smaller or equal to **r** unit.

As you are really happy with the deal, you want to buy all the magnets available in the store by spending as little money as possible. You just have to figure out the smallest bag of size **X** that you have to buy.

Input

Input starts with an integer **T** (≤ 150), denoting the number of test cases. Each case starts with an integer **N** ($1 \le N \le 100$), the number of magnets available at the shop. Followed by **N** integers, R_0, R_1, R_2 ... R_{N-1}, where R_i ($1 \le R_i \le 100$) is the radius of i^{th} magnet.

Output

For each test case, print the case number and the minimum size of the bag you have to buy.

Sample Input	Output for Sample Input
2 3 1 2 3 2 3 3	Case 1: 3 Case 2: 3

Explanation

Case 1: As the magnets have radius of 1, 2, 3 respectively, you can buy a bag of size 3 (which can carry magnets that have a radius smaller or equal to 3) and thus buy all the magnets in the shop.

0/3 BDOSN Toph β

nsu acm-w

Problem B

Amelia's Map

Input: Standard Input
Output: Standard Output

Once upon a time, a girl called Amelia saved enough money to buy a yellow airplane. She called it The Canary. A few years later, she became the first woman to fly solo across the Atlantic Ocean. It was a dangerous flight. Her tiny plane was tossed around by strong winds and icy storms. She kept herself going with a can of tomato juice, sucked through a straw. After almost fifteen hours she touched down in a field in Northern Ireland, much to the surprise of the cows. "Have you come far?" the farmer asked her. "All the way from America" she laughed. - Good Night Stories for Rebel Girls

On the way to Ireland, Amelia's map got tore down into **N** pieces. The torn pieces lost their order and Amelia alone couldn't sort them out. So the farmer and the cows decided to help her.

Each cow picked up one piece of the map and after discussing with each other, they decided which piece will come in what order. The farmer told to line themselves up according to their piece number. But you know cows. They are never where they should be. So they came to Amelia in a messed up order.

Whenever a cow comes to Amelia, she checks the piece number and only if all the previous pieces of the map are already attached to the map, she accepts the piece, attaches it to the map and sends the cow out of the line. Otherwise, she sends the cow at the end of the line. The cows keep on coming until the map is finished. **Note that, she can only start remaking her map only when she gets the 1st piece.**

For example, if the cows come in the order [4, 1, 3, 2] then Amelia first receives the 4th piece of map, which she cannot accept as she has not found the 1st piece yet. So she sends the cow back and array becomes [1, 3, 2, 4]. Next, Amelia receives 1, which she accepts and attached to the map. She sends this cow out of the line so array becomes [3, 2, 4]. Next she receives 3, but she cannot accept this piece as only the first piece has been attached to map. She sends the cow back and array becomes [2, 4, 3]. Next she receives the 2nd piece which she happily accepts since the first piece has been attached. So the array becomes [4, 3]. She cannot accept 4th piece yet since 3rd piece has not been attached. So array becomes [3, 4]. Next she accepts 3rd and then 4th piece.

The whole process is very time-consuming. Processing one request takes 1 second. Now, as you are Amelia's renowned programmer friend, she asks you to count the total time to finish the map, given the order of the pieces.

But again, the cows are huge fans of cryptography! They just wouldn't give Amelia the permutation order they came in to her. Instead they handed her an initial permutation **A** of length **L** and every time Amelia loses her map, they provide her an integer **k**, and Amelia is to generate the original order from this function -

BD SN Toph β

Prob-B

```
/* Implementation of the generating function using C */
void input_generator(int A[], int L, int N, int k, int input[]) {
    for(int i = 0, j = 0; i < L; i ++) {
        int x = A[i] ^ k;
        if(x < N) input[j ++] = x + 1;
    }
}
```

After that first **N** numbers of **input** array will contain the original order the **N** cows came in. **It is guaranteed that input will contain a valid permutation of 1 to N.**

Input

Input starts with an integer **L** (**1 <= L <= 2^{20}**) and the initial permutation **A** (**0 <= A[i] < L**).
Next line contains an integer **T** (**1 <= T <= 100**), the number of test cases (the number of times Amelia lost her map).

Each case contains two integers **N** (**1 <= N <= L**) and **k** (**0 <= k < L**), the number of pieces the map got torn into and the number Amelia gets to pass to the generator function.

Output

For each case, print the case number and the total number of seconds to finish the map.

Sample Input	Output for Sample Input
8 3 0 6 1 7 2 4 5 3 1 5 3 4 7 3	Case 1: 1 Case 2: 4 Case 3: 18

Explanation

First sample
After passing A, N and k to inpute_generator function, input array of length 1 will be : 1
First cow comes with 1st piece, Amelia accepts it → 1 second
Total time = 1 second

Second sample
Input array of length 3 will be : 3 1 2
First cow comes with 3rd piece, Amelia ignores and sends the cow at the end of the line → 1 second
Second cow comes with 1st piece, Amelia accepts it → 1 second
Third cow comes with 2nd piece, Amelia accepts it → 1 second
Then, the first cow comes with 3rd piece again, Amelia accepts it now → 1 second
Total time = 1 + 1 + 1 + 1 = 4 seconds

BD SN Toph β

nsu acm-w

Problem C

Kylo Ren's Puzzle

Input: Standard Input
Output: Standard Output

In a galaxy far far away, there is a galactic war going on. The evil First Order is fighting the Rebels. In his attempt to finish off the Rebels, the supreme leader of the First Order, Kylo Ren, has captured the last Jedi, Rey. She is being kept in a cell with you. It is her destiny to escape from this cell and defeat Kylo Ren with her mastery of the Force. But in order to open the cell gates, she must solve a puzzle prepared by Kylo himself. The puzzle goes like this:

You are given a **tree** (an undirected acyclic graph) with **N** nodes. The nodes are numbered from **1** to **N**. The edges of the tree are **unweighted** (weight of each edge is 1). For each node, you have to print the sum of distance from that node to all other nodes.

Since you are good with programming, help her solve this problem and fulfil her destiny.

Input

The first line of input contains an integer **T** (**1 <= T <= 10**), the number of test cases. **T** test cases follow.

The first line of each test case contains an integer **N** (**1 <= N <= 100000**), the number of nodes in the tree. The next **N-1** lines contain two integers each. The i^{th} of these lines contains integers u_i and v_i (**1 <= u_i, v_i <= N, u_i != v_i**), representing that there is an edge between the nodes u_i and v_i.

You can safely assume that the given graph will form a valid tree, i.e. between any two nodes, there exists exactly one simple path.

Output

For each case, print **N** space separated integers in a separate line. The i^{th} of these integers represents the sum of distance from the i^{th} node to all other nodes.

Sample Input	Output for Sample Input
2 5 1 2 2 3 3 4 4 5 3 3 2 2 1	10 7 6 7 10 3 2 3

BD SN Toph β

Explanation

Tree in the first case looks like the following.

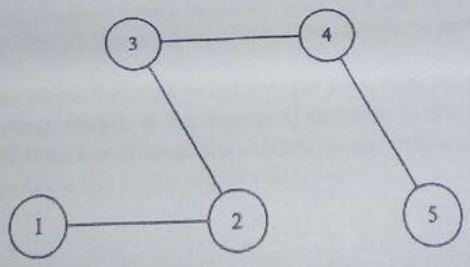
Distance of the nodes from node 1 is 0 (Node 1 itself), 1 (Node 2), 2 (Node 3), 3 (Node 4) and 4 (Node 5). So the sum of the distances is 10.

Distance of the nodes from node 2 is 1 (Node 1), 0 (Node 2 itself), 1 (Node 3), 2 (Node 4) and 3 (Node 5). So the sum of the distances is 7.

Distance of the nodes from node 3 is 2 (Node 1), 1 (Node 2), 0 (Node 3 itself), 1 (Node 4) and 2 (Node 5). So the sum of the distances is 6.

Distance of the nodes from node 4 is 3 (Node 1), 2 (Node 2), 1 (Node 3), 0 (Node 4 itself) and 1 (Node 5). So the sum of the distances is 7.

Distance of the nodes from node 5 is 4 (Node 1), 3 (Node 2), 2 (Node 3), 1 (Node 4) and 0 (Node 5 itself). So the sum of the distances is 10.



9/43

BD SN Toph β

Problem D

Input: Standard Input
Output: Standard Output

The Order of the Queen

Razia Sultana, the renowned Sultan of Delhi, was busy preparing her troops for a battle against invaders from the North. She decided to align all her soldiers in a single row, and ideally she would want the height of all the soldiers to be in non-decreasing order from left to right. However, once she ordered her troops to align themselves, they arranged themselves hurriedly in an arbitrary order.

The Sultana has **n** soldiers in her army, and she decides to number them **1 to n** from left to right. She calls a pair **(i, j)** a mismatched pair if **i^{th}** soldier is strictly taller than the **j^{th}** soldier where **i<j**.

Now, given the height of each soldier in the line, the Sultana wants to know the answers to **q** questions, and you, as her trusted second-in-command need to answer them fast!

Each question consists of two integers **L** and **R (L<R)**, and you need to count the number of mismatched pairs within the range **L to R** (inclusive).

Input

The first line of input consists of a single integer **T (1<=T<=10)**, denoting the number of test cases. Each test case starts with a number **n (1<=n<=10000)**, denoting the number of soldiers. The following line contains **n** space separated integers, the **i^{th}** integer denoting **h_i(1<=h_i<=10)**, the height of the **i^{th}** soldier in the line.

Then follows a line containing **q (1<=q<=100)**, the number of questions the queen will ask, followed by **q** lines, containing two integers each, denoting the values of **L** and **R (1<=L<R<=n)** for the question.

Output

For each question, print a single integer denoting the answer to the queen's question.

Sample Input	Output for Sample Input
1 5 1 9 2 7 8 3 1 3 2 4 1 5	1 2 3

Prob
-D

BD SN Toph β

Explanation

In the sample case, the soldiers in line have heights 1, 9, 2, 7, 8 from left to right.

Considering only the 1st, 2nd and 3rd soldier, there is only one mismatched pair, the 2nd and the 3rd soldier. Because the height of the 2nd soldier is 9, which is greater than that of the third soldier, 2. Hence the answer to the first question is 1. Notice that pairs (1, 2) and (1, 3) are not mismatched pairs because the height of the 1st soldier is less than those of the 2nd and 3rd soldiers respectively. Also notice that although (2, 4) is a mismatched pair, it is not counted in the answer to this question because the 4th soldier is not within the given range 1-3.

There are 2 mismatched pairs between soldiers 2-4, the pairs (2, 3) and (2, 4). Thus the answer to the second question is 2.

The 3rd question asks you to find the number of mismatched pairs in the range 1-5. There are 3 mismatched pairs, (2, 3), (2, 4) and (2, 5), all of which are within the given range.

2/9 BDOSN Toph β

Problem E

Input: Standard Input
Output: Standard Output

Arju and Ayush with Lucky numbers

Arju and Ayush are siblings. Arju loves lucky numbers. We know that positive integers are **lucky** if their decimal representation does not contain digits other than 4 and 7. For example, numbers 47, 744, 4 are lucky but 5, 17, 467 are not.

One day Ayush gave a range of positive numbers from **L** to **R** (**L<=R**) to Arju. Arju should find following.

1. Write all lucky numbers from L to R sequentially in increasing order. It is an array.
2. Find FUN of that array. FUN of an array (arr) of length L is,
   - If array length (L) is zero then FUN is -1.
   - Otherwise,

$$FUN = \sum_{i=1}^{L-1} (arr[i+1] - arr[i])^2$$

Arju is so lazy to solve this problem and wants help from you.

Input

First line of input contains a number **Q($1<=Q<=10^5$)** denoting number of ranges given. Next Q lines contains two integers **L and R ($1<=L<=R<=10^{18}$)** denoting the range.

Output

For each range, if FUN value is -1 then just print -1, else print FUN modulo 1000000007.

Sample Input	Output for Sample Input
7 1 100 3 87 4 7 12 43 44 44 1 1000000000000000000 777777777777777778 777777777777777779	2125 2125 9 -1 0 307881952 -1

Explanation

For the first query, the array will be {4, 7, 44, 47, 74, 77}
FUN=$(7-4)^2+(44-7)^2+(47-44)^2+(74-47)^2+(77-74)^2$=2125

50/52 BDOSN Toph β

Problem F

Input: Standard Input
Output: Standard Output

My Dear J++ !!!

Meena and Raju recently learned about C++. They think STL (Standard template library) of C++ made their life simple and easy than ever before (in terms of coding :P). So, they become very much interested in learning the new programming language "J++".

J++ is a case-sensitive language. J++ has a naming convention. The J++ naming convention is a rule that guides us how to name variables and constants. There are also some keywords in J++.

Keywords are predefined, reserved and have special meanings to the compiler. **auto**, **extern**, **return**, **void**, **while** , **goto.** These are the **six** keywords that are reserved in J++. You cannot use them as variable or constant names.

Variable name: Variable names should consist of the lowercase Latin letters only.
Constant name: Constant names should consist of the uppercase Latin letters only.

These rules are little complicated, isn't it? As Meena and Raju are not familiar with J++ you need to help them. In this program, you will be given a non-empty string. You have to find whether it is a keyword, variable name or constant name.

Input

Input starts with an integer **T (<=100)** denoting the number of test cases. Following **T** line will contain a non-empty string **S (|S|<=20)**. You may safely assume that **S** will consist of lowercase and uppercase Latin letters only.

Output

For each test case print the case number followed by the type of the string.

If given string is a keyword print "Keyword".
If given string is a Variable print "Variable".
If given string is a Constant print "Constant".
If you are not sure about it print "My Dear J++!!!".

Sample Input	Output for Sample Input
5 RED auto CHAR a AreYouSure	Case 1: Constant Case 2: Keyword Case 3: Constant Case 4: Variable Case 5: My Dear J++!!!

4/15 BD SN Toph β

Input: Standard Input
Output: Standard Output

Problem G

Median Game

One day Arya and Bran, the two little kids of Stark family were playing at the seashore. Suddenly they found a beautiful marvel on the beach. The marvel was so astonishing that they both wanted to have it at any cost and so they started quarreling with each other. As they couldn't reach out to any agreement by themselves, they went to their elder sister Sansa Stark for help. Sansa is a very clever girl. She instantly found a nice solution. As both Arya and Bran are very good in programming, she gave them a very interesting problem to solve. Whoever comes with the correct solution first, will acquire that marvel as a gift. But Sansa also doesn't know the solution of the problem yet. And she's little busy right now with preparing the foods. So she has asked you to solve that problem for her so that she can justify their solutions.

You will be given two arrays **A** and **B** of length **N** and **M**. You can add at max **K** new elements of any value into the array **A**. You have to minimize the difference between the median of the two arrays **A** and **B**.

More specifically, you are allowed to add at max **K** new elements into the array **A** but you can not make any change in array **B**. After adding those elements let's say now the median of array **A** is **m1** and the median of array **B** is **m2**. You need to minimize the difference **|m1-m2|**.

Median *of an array is the element at position* **L/2** *(Integer division) in ascending order of the elements (following 0-based indexing).* Here **L** is the number of elements in the array. For example Let's say array **A** = *[2, 5, 3, 1, 4, 6]*. Now if we arrange the numbers in ascending order then **A** will look like this: *[1, 2, 3, 4, 5, 6]*. So median of **A** is A[L/2] = A[6/2] = A[3] = 4.

Input

The first line contains an integer **T**(1<=T<=50) the number of test cases. The first line of each test case contains **3** integers **N**($1<=N<=10^5$), **M**($1<= M<=10^5$) and **K**($0<=K<=10^5$). Second line of each test case contains **N** space separated integers representing array **A** and third line of test case contains **M** space separated integer representing array **B** ($-10^9<= A_i, B_i <=10^9$).

Sum of N over all test cases is <= 10^6 and sum of M over all test cases is <= 10^6.

Output

For each test case print a single integer, the minimum difference between the median of those two arrays.

nsu Problem H

BD SN Toph β

Prob-G

Sample Input	Output for Sample Input
2 5 5 3 3 15 20 8 10 50 25 7 35 30 5 3 2 20 4 2 8 25 1 2 3	Case 1: 10 Case 2: 2

Explanation

Test Case 1

Let's add 3 new elements 35, 40 and 50 into array A and then sort both array A and B.

Now A looks like this: [3, 8, 10, 15, 20, 35, 40, 50] and B looks like this: [7, 25, 30, 35, 50]

The median of A is m1 = 20 and median of B is m2 = 30. (0 based index). Hence the answer is |m1-m2| = 10.

50/58 BD SN Toph β

nsu acm-W

Problem H

Input: Standard Input
Output: Standard Output

Prime Difference

You will be given an integer **N**(2 <= **N** <= 100000). You have to find two prime numbers **p** and **q** such that 1 < **p** <= **q**<= **N** and the difference between **p** and **q** is maximized.

Input

First line of the input is **T**(<=100), then T test cases follows. Each case have only one line containing a positive integers **N** (2 <= **N** <= 100000).

Output

For each test case, output a line containing "Case I: d" where **I** is test case number and **d** is the maximum value of **q - p**, that is maximum difference between two primes, both less than or equal **N**.

Sample Input	Output for Sample Input
2 4 5	Case 1: 1 Case 2: 3

Hint

This problem can be solved using multiple ways. You can use the following algorithm to solve the problem but it is not strictly necessary.

- 1 is not a prime.
- All primes except 2 are odd.
- All prime numbers greater than 3 can be written as 6k+1 or 6k-1.
- Any number n can have only one prime factor greater than $\sqrt{n}$.
- The consequence for primality testing of a number n is: if we cannot find a number f less than or equal $\sqrt{n}$ that divides n then n is prime.

You can check whether a number is prime or not by using the following c++ code.

```
bool isPrime(int n) {
  if (n <= 1)  return false; if (n <= 3)  return true;   // Corner cases
  // This is checked so that we can skip middle five numbers in below loop
  if (n%2 == 0 || n%3 == 0) return false
  for (int i=5; i*i<=n; i=i+6)
    if (n%i == 0 || n%(i+2) == 0)
      return false
  return true;
}
```

0/1

BD SN Toph β

Input: Standard Input
Output: Standard Output

Problem I

Winning Probability

Alice decided to play pillow passing game with her **N** friends on her birthday. In the game **N** people seat in a circle and are numbered from **1** to **N**. Initially, the *1st* person has the pillow. When a song begins, the *1st* person passes the pillow to the *2nd* person, then *2nd* person to *3rd* person, ... *(N-1)th* person to *Nth* person, *Nth* person to *1st* person and so on until the song ends. At the end of the song, the person who holds the pillow loses that round and leaves the game. Next round begins with the next person. Here we consider that passing a pillow to next person takes 1 second. After *N-1* rounds, the last person to survive wins the game.

Alice's best friend Bob already knows about the number of songs **S** and their length **L**, but does not know which song will be played on which round. Bob is seating at *Xth* position and wants to know what are his chances of winning the game. We can assume that all songs have an equal probability of being chosen for any round and the same song can be reused in multiple rounds.

Input

The first line contains an integer representing number of test cases **T(1 <= T <= 100)**. Each test case is represented by two lines. The first line contains three integers representing the number of friends **N(2 <= N <= 100)**, the number of songs **S(1 <= S <= 100)** and the position of Bob **X(1 <= X <= N)**. And the second line contains **S** integers **($1 <= S_i <= 10^5$)**, where *ith* number represents the length of the *ith* song.

Output

For each test case print the answer in a single line. Your answer will be considered correct if its absolute error does not exceed **1e-6**. Formally, the answer is correct if **|x-y| <= 1e-6** where x is judge's answer and y is yours.

Sample Input	Output for Sample Input
1 3 2 2 5 3	0.50000000000

Explanation

Alice can play music in four possible ways:
1. Play the first song (duration = 5) and eliminate the third person. The next round starts again from the first person with the same song. At last, Bob eliminates and lose the game.
2. Play the first song (duration = 5) and eliminate the third person. The next round starts again from the first person with the second song (duration = 3). At last, Bob eliminates and lose the game.
3. Play the second song (duration = 3) and eliminate the first person. The next round starts from Bob with the first song (duration = 5). At last, the other person eliminates and Bob wins the game.
4. Play the second song (duration = 3) and eliminate the first person. The next round starts from Bob with the same song. At last, the other person eliminates and Bob wins the game.

Since out of four games Bob wins two, the probability is half.