GPC-UPC Divide and Conquer Contest

A. Imbalanced Array

2 seconds, 256 megabytes

You are given an array a consisting of n elements. The *imbalance value* of some subsegment of this array is the difference between the maximum and minimum element from this segment. The *imbalance value* of the array is the sum of *imbalance values* of all subsegments of this array.

For example, the *imbalance value* of array $[1, 4, 1]$ is 9, because there are 6 different subsegments of this array:

- $[1]$ (from index 1 to index 1), *imbalance value* is 0;
- $[1, 4]$ (from index 1 to index 2), *imbalance value* is 3;
- $[1, 4, 1]$ (from index 1 to index 3), *imbalance value* is 3;
- $[4]$ (from index 2 to index 2), *imbalance value* is 0;
- $[4, 1]$ (from index 2 to index 3), *imbalance value* is 3;
- $[1]$ (from index 3 to index 3), *imbalance value* is 0;

You have to determine the *imbalance value* of the array a.

Input
The first line contains one integer n ($1 \le n \le 10^6$) — size of the array a.

The second line contains n integers $a_1, a_2 \dots a_n$ ($1 \le a_i \le 10^6$) — elements of the array.

Output
Print one integer — the *imbalance value* of a.

input
3 1 4 1
output
9

B. Pashmak and Parmida's problem

3 seconds, 256 megabytes

Parmida is a clever girl and she wants to participate in Olympiads this year. Of course she wants her partner to be clever too (although he's not)! Parmida has prepared the following test problem for Pashmak.

There is a sequence a that consists of n integers $a_1, a_2, \dots, a_n$. Let's denote $f(l, r, x)$ the number of indices k such that: $l \le k \le r$ and $a_k = x$. His task is to calculate the number of pairs of indicies i, j ($1 \le i < j \le n$) such that $f(1, i, a_i) > f(j, n, a_j)$.

Help Pashmak with the test.

Input
The first line of the input contains an integer n ($1 \le n \le 10^6$). The second line contains n space-separated integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^9$).

Output
Print a single integer — the answer to the problem.

input
7 1 2 1 1 2 2 1
output
8

input
3 1 1 1
output
1

input
5 1 2 3 4 5
output
0

C. Equivalent Strings

2 seconds, 256 megabytes

Today on a lecture about strings Gerald learned a new definition of string equivalency. Two strings a and b of equal length are called *equivalent* in one of the two cases:

1. They are equal.
2. If we split string a into two halves of the same size a_1 and a_2, and string b into two halves of the same size b_1 and b_2, then one of the following is correct:
   a. a_1 is equivalent to b_1, and a_2 is equivalent to b_2
   b. a_1 is equivalent to b_2, and a_2 is equivalent to b_1

As a home task, the teacher gave two strings to his students and asked to determine if they are equivalent.

Gerald has already completed this home task. Now it's your turn!

Input
The first two lines of the input contain two strings given by the teacher. Each of them has the length from 1 to $200\,000$ and consists of lowercase English letters. The strings have the same length.

Output
Print "YES" (without the quotes), if these two strings are equivalent, and "NO" (without the quotes) otherwise.

input
aaba abaa
output
YES

input
aabb abab
output
NO

In the first sample you should split the first string into strings "aa" and "ba", the second one — into strings "ab" and "aa". "aa" is equivalent to "aa"; "ab" is equivalent to "ba" as "ab" = "a" + "b", "ba" = "b" + "a".

In the second sample the first string can be splitted into strings "aa" and "bb", that are equivalent only to themselves. That's why string "aabb" is equivalent only to itself and to string "bbaa".

D. The Brand New Function

2 seconds, 256 megabytes

Polycarpus has a sequence, consisting of n non-negative integers: $a_1, a_2, \dots, a_n$.

Let's define function $f(l, r)$ (l, r are integer, $1 \le l \le r \le n$) for sequence a as an operation of bitwise OR of all the sequence elements with indexes from l to r. Formally: $f(l, r) = a_l \mid a_{l+1} \mid \dots \mid a_r$.

Polycarpus took a piece of paper and wrote out the values of function $f(l, r)$ for all l, r (l, r are integer, $1 \le l \le r \le n$). Now he wants to know, how many **distinct** values he's got in the end.

Help Polycarpus, count the number of distinct values of function $f(l, r)$ for the given sequence a.

Expression $x \mid y$ means applying the operation of bitwise OR to numbers x and y. This operation exists in all modern programming languages, for example, in language *C++* and *Java* it is marked as "|", in *Pascal* — as "or".

Input
The first line contains integer n ($1 \le n \le 10^5$) — the number of elements of sequence a. The second line contains n space-separated integers $a_1, a_2, ..., a_n$ ($0 \le a_i \le 10^6$) — the elements of sequence a.

Output
Print a single integer — the number of distinct values of function $f(l, r)$ for the given sequence a.

Please, do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use `cin`, `cout` streams or the `%I64d` specifier.

input
3 1 2 0

output
4

input
10 1 2 3 4 5 6 1 2 9 10

output
11

In the first test case Polycarpus will have 6 numbers written on the paper: $f(1, 1) = 1$, $f(1, 2) = 3$, $f(1, 3) = 3$, $f(2, 2) = 2$, $f(2, 3) = 2$, $f(3, 3) = 0$. There are exactly 4 distinct numbers among them: $0, 1, 2, 3$.

E. Package Delivery

2 seconds, 256 megabytes

Johnny drives a truck and must deliver a package from his hometown to the district center. His hometown is located at point 0 on a number line, and the district center is located at the point d.

Johnny's truck has a gas tank that holds exactly n liters, and his tank is initially full. As he drives, the truck consumes exactly one liter per unit distance traveled. Moreover, there are m gas stations located at various points along the way to the district center. The i-th station is located at the point x_i on the number line and sells an unlimited amount of fuel at a price of p_i dollars per liter. Find the minimum cost Johnny must pay for fuel to successfully complete the delivery.

Input
The first line of input contains three space separated integers d, n, and m ($1 \le n \le d \le 10^9$, $1 \le m \le 200\,000$) — the total distance to the district center, the volume of the gas tank, and the number of gas stations, respectively.

Each of the next m lines contains two integers x_i, p_i ($1 \le x_i \le d - 1$, $1 \le p_i \le 10^6$) — the position and cost of gas at the i-th gas station. It is guaranteed that the positions of the gas stations are distinct.

Output
Print a single integer — the minimum cost to complete the delivery. If there is no way to complete the delivery, print `-1`.

input
10 4 4 3 5 5 8 6 3 8 4

output
22

input
16 5 2 8 2 5 1

output
-1

In the first sample, Johnny's truck holds 4 liters. He can drive 3 units to the first gas station, buy 2 liters of gas there (bringing the tank to 3 liters total), drive 3 more units to the third gas station, buy 4 liters there to fill up his tank, and then drive straight to the district center. His total cost is $2 \cdot 5 + 4 \cdot 3 = 22$ dollars.

In the second sample, there is no way for Johnny to make it to the district center, as his tank cannot hold enough gas to take him from the latest gas station to the district center.

F. Maximum Median

2 seconds, 256 megabytes

You are given an array a of n integers, where n is odd. You can make the following operation with it:

- Choose one of the elements of the array (for example a_i) and increase it by 1 (that is, replace it with $a_i + 1$).

You want to make the median of the array the largest possible using at most k operations.

The median of the odd-sized array is the middle element after the array is sorted in non-decreasing order. For example, the median of the array $[1, 5, 2, 3, 5]$ is 3.

Input
The first line contains two integers n and k ($1 \le n \le 2 \cdot 10^5$, n is odd, $1 \le k \le 10^9$) — the number of elements in the array and the largest number of operations you can make.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$).

Output
Print a single integer — the maximum possible median after the operations.

input
3 2 1 3 5

output
5

input
5 5 1 2 1 1 1

output
3

input
7 7 4 1 2 4 3 4 4

output
5

In the first example, you can increase the second element twice. Than array will be $[1, 5, 5]$ and it's median is 5.

In the second example, it is optimal to increase the second number and than increase third and fifth. This way the answer is 3.

In the third example, you can make four operations: increase first, fourth, sixth, seventh element. This way the array will be $[5, 1, 2, 5, 3, 5, 5]$ and the median will be 5.

G. Creative Snap

1 second, 256 megabytes

Thanos wants to destroy the avengers base, but he needs to destroy the avengers along with their base.

Let we represent their base with an array, where each position can be occupied by many avengers, but one avenger can occupy only one position. Length of their base is a perfect power of 2. Thanos wants to destroy the base using minimum power. He starts with the whole base and in one step he can do either of following:

- if the current length is at least 2, divide the base into 2 equal halves and destroy them separately, or
- burn the current base. If it contains no avenger in it, it takes A amount of power, otherwise it takes his $B \cdot n_a \cdot l$ amount of power, where n_a is the number of avengers and l is the length of the current base.

Output the minimum power needed by Thanos to destroy the avengers' base.

Input

The first line contains four integers n, k, A and B ($1 \le n \le 30$, $1 \le k \le 10^5$, $1 \le A, B \le 10^4$), where 2^n is the length of the base, k is the number of avengers and A and B are the constants explained in the question.

The second line contains k integers $a_1, a_2, a_3, \ldots, a_k$ ($1 \le a_i \le 2^n$), where a_i represents the position of avenger in the base.

Output

Output one integer — the minimum power needed to destroy the avengers base.

input
2 2 1 2 1 3
output
6

input
3 2 1 2 1 7
output
8

Consider the first example.

One option for Thanos is to burn the whole base $1 - 4$ with power $2 \cdot 2 \cdot 4 = 16$.

Otherwise he can divide the base into two parts $1 - 2$ and $3 - 4$.

For base $1 - 2$, he can either burn it with power $2 \cdot 1 \cdot 2 = 4$ or divide it into 2 parts $1 - 1$ and $2 - 2$.

For base $1 - 1$, he can burn it with power $2 \cdot 1 \cdot 1 = 2$. For $2 - 2$, he can destroy it with power 1, as there are no avengers. So, the total power for destroying $1 - 2$ is $2 + 1 = 3$, which is less than 4.

Similarly, he needs 3 power to destroy $3 - 4$. The total minimum power needed is 6.

H. Petya and Array

2 seconds, 256 megabytes

Petya has an array a consisting of n integers. He has learned partial sums recently, and now he can calculate the sum of elements on any segment of the array really fast. The segment is a non-empty sequence of elements standing one next to another in the array.

Now he wonders what is the number of segments in his array with the sum less than t. Help Petya to calculate this number.

More formally, you are required to calculate the number of pairs l, r ($l \le r$) such that $a_l + a_{l+1} + \cdots + a_{r-1} + a_r < t$.

Input

The first line contains two integers n and t ($1 \le n \le 200\,000, |t| \le 2 \cdot 10^{14}$).

The second line contains a sequence of integers $a_1, a_2, \ldots, a_n$ ($|a_i| \le 10^9$) — the description of Petya's array. Note that there might be negative, zero and positive elements.

Output

Print the number of segments in Petya's array with the sum of elements less than t.

input
5 4 5 -1 3 4 -1
output
5

input
3 0 -1 2 -3
output
4

input
4 -1 -2 1 -2 3
output
3

In the first example the following segments have sum less than 4:

- $[2, 2]$, sum of elements is -1
- $[2, 3]$, sum of elements is 2
- $[3, 3]$, sum of elements is 3
- $[4, 5]$, sum of elements is 3
- $[5, 5]$, sum of elements is -1

I. Kefa and Company

2 seconds, 256 megabytes

Kefa wants to celebrate his first big salary by going to restaurant. However, he needs company.

Kefa has n friends, each friend will agree to go to the restaurant if Kefa asks. Each friend is characterized by the amount of money he has and the friendship factor in respect to Kefa. The parrot doesn't want any friend to feel poor compared to somebody else in the company (Kefa doesn't count). A friend feels poor if in the company there is someone who has at least d units of money more than he does. Also, Kefa wants the total friendship factor of the members of the company to be maximum. Help him invite an optimal company!

Input

The first line of the input contains two space-separated integers, n and d ($1 \le n \le 10^5, 1 \le d \le 10^9$) — the number of Kefa's friends and the minimum difference between the amount of money in order to feel poor, respectively.

Next n lines contain the descriptions of Kefa's friends, the $(i+1)$-th line contains the description of the i-th friend of type m_i, s_i ($0 \le m_i, s_i \le 10^9$) — the amount of money and the friendship factor, respectively.

Output
Print the maximum total friendship factir that can be reached.

input
4 5 75 5 0 100 150 20 75 1
output
100

input
5 100 0 7 11 32 99 10 46 8 87 54
output
111

In the first sample test the most profitable strategy is to form a company from only the second friend. At all other variants the total degree of friendship will be worse.

In the second sample test we can take all the friends.

J. Burning Midnight Oil

2 seconds, 256 megabytes

One day a highly important task was commissioned to Vasya — writing a program in a night. The program consists of n lines of code. Vasya is already exhausted, so he works like that: first he writes v lines of code, drinks a cup of tea, then he writes as much as $\lfloor \frac{v}{k} \rfloor$ lines, drinks another cup of tea, then he writes $\lfloor \frac{v}{k^2} \rfloor$ lines and so on: $\lfloor \frac{v}{k^3} \rfloor, \lfloor \frac{v}{k^4} \rfloor, \lfloor \frac{v}{k^5} \rfloor$, ...

The expression $\lfloor \frac{a}{b} \rfloor$ is regarded as the integral part from dividing number a by number b.

The moment the current value $\lfloor \frac{v}{k^p} \rfloor$ equals 0, Vasya immediately falls asleep and he wakes up only in the morning, when the program should already be finished.

Vasya is wondering, what minimum allowable value v can take to let him write **not less** than n lines of code before he falls asleep.

Input
The input consists of two integers n and k, separated by spaces — the size of the program in lines and the productivity reduction coefficient, $1 \le n \le 10^9$, $2 \le k \le 10$.

Output
Print the only integer — the minimum value of v that lets Vasya write the program in one night.

input
7 2
output
4

input
59 9
output
54

In the first sample the answer is $v = 4$. Vasya writes the code in the following portions: first 4 lines, then 2, then 1, and then Vasya falls asleep. Thus, he manages to write $4 + 2 + 1 = 7$ lines in a night and complete the task.

In the second sample the answer is $v = 54$. Vasya writes the code in the following portions: 54, 6. The total sum is $54 + 6 = 60$, that's even more than $n = 59$.

K. Queue

2 seconds, 256 megabytes

There are n walruses standing in a queue in an airport. They are numbered starting from the queue's tail: the 1-st walrus stands at the end of the queue and the n-th walrus stands at the beginning of the queue. The i-th walrus has the age equal to a_i.

The i-th walrus becomes displeased if there's a younger walrus standing in front of him, that is, if exists such j ($i < j$), that $a_i > a_j$. The displeasure of the i-th walrus is equal to the number of walruses between him and the furthest walrus ahead of him, which is younger than the i-th one. That is, the further that young walrus stands from him, the stronger the displeasure is.

The airport manager asked you to count for each of n walruses in the queue his displeasure.

Input
The first line contains an integer n ($2 \le n \le 10^5$) — the number of walruses in the queue. The second line contains integers a_i ($1 \le a_i \le 10^9$).

Note that some walruses can have the same age but for the displeasure to emerge the walrus that is closer to the head of the queue needs to be **strictly younger** than the other one.

Output
Print n numbers: if the i-th walrus is pleased with everything, print "-1" (without the quotes). Otherwise, print the i-th walrus's displeasure: the number of other walruses that stand between him and the furthest from him younger walrus.

input
6 10 8 5 3 50 45
output
2 1 0 -1 0 -1

input
7 10 4 6 3 2 8 15
output
4 2 1 0 -1 -1 -1

input
5 10 3 1 10 11
output
1 0 -1 -1 -1

L. Three Sorted Arrays

2 seconds, 512 megabytes

This time Laltu wanted the question to be straightforward. So, given 3 1-indexed sorted arrays ($A[], B[], C[]$), find the number of triplets $1 \le i \le j \le k$, such that: $A[i] \le B[j] \le C[k]$. Note that i, j and k don't exceed the size of respective arrays.

For example, Arrays: $A = [1, 2, 3, 4]$ $B = [5, 6, 7, 8]$ $C = [9, 10, 11, 12]$

The triplet $(i, j, k) = (1, 2, 3)$ has to be considered because: $A[1] \leq B[2] \leq C[3]$.

Input
First line contains T, the number of test cases. Each test case consists of: P, the length of first array. The next line will consist of P integers. Q, the length of second array. The next line will consist of Q integers. R, the length of third array. The next line will consist of R integers.

Output
For each test case print the required answer in one line.

Constraints

- $1 \leq T \leq 3$
- $1 \leq P, Q, R \leq 10^5$
- $-10^9 \leq$ Elements of arrays $\leq 10^9$

input
1 3 1 5 6 3 2 3 4 3 7 8 9
output
6

The possible triplets (i, j, k) are: (1, 1, 1) (1, 1, 2) (1, 1, 3) (1, 2, 2) (1, 2, 3) (1, 3, 3)

M. Points on Line

2 seconds, 256 megabytes

Little Petya likes points a lot. Recently his mom has presented him n points lying on the line OX. Now Petya is wondering in how many ways he can choose three distinct points so that the distance between the two farthest of them doesn't exceed d.

Note that the order of the points inside the group of three chosen points doesn't matter.

Input
The first line contains two integers: n and d ($1 \leq n \leq 10^5$; $1 \leq d \leq 10^9$). The next line contains n integers $x_1, x_2, ..., x_n$, their absolute value doesn't exceed 10^9 — the x-coordinates of the points that Petya has got.

It is guaranteed that the coordinates of the points in the input **strictly increase**.

Output
Print a single integer — the number of groups of three points, where the distance between two farthest points doesn't exceed d.

Please do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

input
4 3 1 2 3 4
output
4

input
4 2 -3 -2 -1 0
output
2

input
5 19 1 10 20 30 50
output
1

In the first sample any group of three points meets our conditions.

In the seconds sample only 2 groups of three points meet our conditions: `{-3, -2, -1}` and `{-2, -1, 0}`.

In the third sample only one group does: `{1, 10, 20}`.

N. Tavas and Karafs

2 seconds, 256 megabytes

Karafs is some kind of vegetable in shape of an $1 \times h$ rectangle. Tavaspolis people love Karafs and they use Karafs in almost any kind of food. Tavas, himself, is crazy about Karafs.

Each Karafs has a positive integer height. Tavas has an infinite **1-based** sequence of Karafses. The height of the i-th Karafs is $s_i = A + (i - 1) \times B$.

For a given m, let's define an m-bite operation as decreasing the height of at most m distinct not eaten Karafses by 1. Karafs is considered as eaten when its height becomes zero.

Now SaDDas asks you n queries. In each query he gives you numbers l, t and m and you should find the largest number r such that $l \leq r$ and sequence $s_l, s_{l+1}, ..., s_r$ can be eaten **by performing m-bite no more than t times** or print -1 if there is no such number r.

Input
The first line of input contains three integers A, B and n ($1 \leq A, B \leq 10^6$, $1 \leq n \leq 10^5$).

Next n lines contain information about queries. i-th line contains integers l, t, m ($1 \leq l, t, m \leq 10^6$) for i-th query.

Output
For each query, print its answer in a single line.

input
2 1 4 1 5 3 3 3 10 7 10 2 6 4 8
output
4 -1 8 -1

input
1 5 2 1 5 10 2 7 4
output
1 2

O. Magic Powder - 2

1 second, 256 megabytes

The term of this problem is the same as the previous one, the only exception — increased restrictions.

Input

The first line contains two positive integers n and k ($1 \le n \le 100\,000$, $1 \le k \le 10^9$) — the number of ingredients and the number of grams of the magic powder.

The second line contains the sequence $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$), where the i-th number is equal to the number of grams of the i-th ingredient, needed to bake one cookie.

The third line contains the sequence $b_1, b_2, \ldots, b_n$ ($1 \le b_i \le 10^9$), where the i-th number is equal to the number of grams of the i-th ingredient, which Apollinaria has.

Output

Print the maximum number of cookies, which Apollinaria will be able to bake using the ingredients that she has and the magic powder.

input

```
1 1000000000
1
1000000000
```

output

```
2000000000
```

input

```
10 1
1000000000 1000000000 1000000000 1000000000 1000000000
1000000000 1000000000 1000000000 1000000000 1000000000
1 1 1 1 1 1 1 1 1 1
```

output

```
0
```

input

```
3 1
2 1 4
11 3 16
```

output

```
4
```

input

```
4 3
4 3 5 6
11 12 14 20
```

output

```
3
```