SoCP Mid-Season Contest

A. How many friends?

1 second, 256 megabytes

Kefka is playing with his friends in a park. There are total n children of them playing. Each child i has some popularity value a_i.Two children are friends with each other if and only if the product of their popularity value is a perfect square. You are required to find the maximum size of group of children in which child is a friend of every other child present in the group.

Input

The first line contains a single integer $t (1 \le t \le 1000)$ — the number of test cases.

The first line of each test case contains a single integers $n(1 \le n \le 2.10^5)$.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n (1 \le a_i \le 10^6)$,the popularity value of each child.

It's guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case print a single integer — the maximum size of the group.

input

```
2
4
2 8 16 4
6
1 9 36 3 27 5
```

output

```
2
3
```

In first case one of the optimal case is a_1 and a_2 forming a group.

In second case the optimal case is a_1 ,a_2 and a_3 forming a group.

B. Crack it

1 second, 256 megabytes

A Malicious Hacker plans to hack Cryptocurrency Wallets. The key of any user consists of lowercase English letters and digits $0 - 1$. For a key to be hackable, letters should be present only in groups of size k. For example, if $k = 2$ the keys "01", "$0ab1$" and "$abab$" are hackable while "$ab0c$" and "abc" are not.

Your task is to find the total number of hackable keys having length between l_1 and l_2 (both inclusive).

Input

The first line of input contains two space seperated integers t and k $(1 \le t, k \le 10^5)$, where t denotes the number of test cases.The description of test cases follows.

The first and only line of each test case contains two space seperated integers l_1 and l_2 $(1 \le l_1 \le l_2 \le 10^5)$.

Output

For each test case print the total number of hackable keys having length between l_1 and l_2 (both inclusive) modulo $10^9 + 7$.

input

```
3 2
1 1
1 2
1 4
```

output

```
2
682
468498
```

input

```
2 4
1 3
1 100000
```

output

```
14
797804073
```

EXPLANATION for **Testset 1**

- For $k = 2$ and $l = 1$, the hackable keys are "0" and "1". $Total\ number =\ 2$

- For $k = 2$ and $l = 2$, the hackable keys can be of 2 types-

1. Containing 2 digits$(0 - 1)$ like "00", "10".

2. Containing a group of 2 letters$(a - z)$ like "ab", "xy".

$Total\ number = \ 4 + 676 = \ 680.$

- For $k = 2$ and $l = 3$, the hackable keys can be of 2 types-

1. Containing 3 digits like "010", "111".

2. Containing a group of 2 letters and a digit like "$ab1$", "$0xy$" but keys like "$a0b$" are not hackable.

$Total\ number = \ 8 + 2704 = \ 2712.$

- For $k = 2$ and $l = 4$, the hackable keys can be of 3 types-

1. Containing 4 digits like "0101", "1111".

2. Containing a group of 2 letters and 2 digits like "$01ab$", "$0xy1$" but keys like "$a01b$" are not hackable.

3. Containing 2 groups of 2 letters each like "$abcd$, "$pqrs$".

$Total\ number = \ 16 + 8112 + 456976 = \ 465104.$

C. Maximize Frequency

1 second, 256 megabytes

Sam loves to play with arrays. Knowing this, Billy gives Sam a puzzle to solve. He gives him an array of positive integers and a number k. Sam can do the following operation at most k times.

Choose any element of the array and increment it by 1.

Sam needs to find out the maximum possible frequency of each element of the array which he can get by performing the above operation at most k times.

See notes for more clarification.

Input

Each test contains multiple test cases. The first line of input contains a single integer $t(1 \leq t \leq 100)$, the number of test cases. Then $2t$ lines follow. The first line for each test case contains two space separated positive integers $n(1 \leq n \leq 10^5)$ and $k(1 \leq k \leq 10^{14})$, the number of elements in the array and maximum number of operations respectively. The next line contains n space separated positive integers, the elements of the array $(1 \leq a_i \leq 10^9)$, where a_i is the i^{th} element.

It is guaranteed that the sum of n over all test cases is at most 10^5.

Output

Print t lines. For each testcase, print n space separated integers $b_i(1 \leq b_i \leq n)$, where b_i is the maximum possible frequency of a_i which Sam can get after performing the operation at most k times

input
```
1
3 2
1 2 1
```
output
```
2 3 2
```

input
```
2
5 10
3 1 5 4 7
3 1
3 1 2
```
output
```
2 1 4 3 4
2 1 2
```

Explanation of first test:

For a_1 = 1, the maximum frequency attainable is 2 ,i.e. by doing 0 operations.

For a_2 = 2, in one operation we can choose a_1 and increase it by 1 such that a_1 = 2. In second operation we can choose a_3 and increase it by 1 such that a_3 = 2. Hence, maximum attainable frequency of a_2 is 3.

For a_3 = 1, the maximum frequency attainable is 2 ,i.e. by doing 0 operations.

D. Worthy Arrays

1 second, 256 megabytes

Shubham is playing a game where he wants to minimize an array's worth.

The worth of an array a is defined as the sum of the absolute value of its elements, $\sum_{i=1}^{n} |a[i]|$.

Shubham can change the values of the array a by applying the following operation arbitrarily many times,

1. Pick two indices i, j such that $|i - j| \leq k$, and $a[i] > 0$. Call i *source* and j *sink*.
2. Pick a positive integer $0 < x \leq a[i]$, decrease $a[i]$ by x and increase $a[j]$ by x.
3. In any subsequent operation, the sink j can no longer be a source.

Find the minimum possible worth of a that Shubham can achieve after performing some (possibly zero) operations.

Input

The first line contains two space separated integers n and k. The next line contains n space separated integers, $a_1, a_2, \ldots, a_n$ representing the array.

$1 \leq n, k \leq 10^5$

$-10^9 \leq a_i \leq 10^9$ for $1 \leq i \leq n$

Output

Print a single integer representing the minimum possible worth.

```
input
3 2
3 -1 -2
output
0
```

```
input
5 2
1 2 3 4 5
output
15
```

In the first testcase, you can turn $[3, -1, -2] \rightarrow [2, 0, -2] \rightarrow [0, 0, 0]$ which has worth 0.

E. Choosy Racers

2 seconds, 256 megabytes

At a racing event, we have N professional racers and C racing cars. Car C_i can be driven only exactly at a speed S_i. Also, all racers have varying levels of experience, for racer R_j we have two-speed limits A_j and B_j ($A_j \leq B_j$). A car with a driving speed less than A_j is too slow for the driver and a car with a speed greater than B_j is too fast. Mathematically, driver R_j can drive car C_i only if $A_j \leq S_i \leq B_j$.

Assuming that each driver can be assigned only one car and each car can be assigned to only one driver, what is the maximum number of drivers that can get to drive in the race?

Input

The first line of input contains C and N.

The next C lines each contain an integer where the i^{th} integer denotes S_i.

The next N lines each contain A_j and B_j ($A_j \leq B_j$) where the j^{th} line denotes speed limits of the j^{th} racer.

$1 \leq C, N \leq 5 * 10^5$

$1 \leq S_i \leq 10^9$

$1 \leq A_j \leq B_j \leq 10^9$

Output

Print a single integer denoting the maximum number of drivers that can get to drive in the race.

```
input
3 3
6
4
7
6 10
8 13
4 8
output
2
```

```
input
4 3
3
17
5
2
5 12
5 9
5 8
```

```
output
1
```

In the first sample test case, we can match C_1 with R_3 and C_3 with R_1 which is the maximum number of pairs we can make, hence the result is 2.

F. Longest Good Subsequence

1 second, 256 megabytes

Given an array $A = [a_1, a_2, \ldots, a_n]$ of n non-negative integers, you need to find the longest good subsequence of the array. If there are multiple good subsequences with the same length(it needs to be longest possible), then you may print any one of them.

A subsequence of an array A is an array B which can be obtained by deleting some(possibly all or none) elements of A and keeping the order of remaining elements unchanged. For eg. if $A = [3, 4, 1, 2, 5]$ then $B = [3, 1, 2]$, $B = []$ and $B = [3, 4, 1, 2, 5]$ are valid subsequences of A but $B = [4, 3]$ and $B = [3, 1, 7]$ are not.

An array $B = [b_1, b_2, \ldots, b_m]$ is a good subsequence of an array $A = [a_1, a_2, \ldots, a_n]$ if both of the following conditions hold:

- B is a subsequence of A
- For all i s.t. $1 \le i < m$, the condition $b_i \ \& \ b_{i+1} = b_i$ holds where $\&$ denotes the bitwise `AND` operation.

Input

The first line contains a single integer n $(1 \le n \le 10^5)$.

The second line contains n space separated integers a_i $(0 \le a_i \le 1000)$ denoting the elements of the array A.

Output

In the first line print a single integer k denoting the length of the possible good subsequence of A.

In the second line, print k space separated integers, the indicies(1-indexed) of the elements of array A which form the longest good subsequence. In case of multiple solutions with same length(longest possible), print any.

```
input
5
1 2 3 4 5
```

```
output
2
1 3
```

```
input
4
2 2 6 2
```

```
output
3
1 2 4
```

For the first testcase, the maximum possible length of a good subsequence is 2. There are many good subsequences with length 2. Some possible answers are : $[1, 3]$, $[1, 5]$, $[2, 3]$ etc.

For the second testcase, there are 2 possible answers: $[1, 2, 3]$ and $[1, 2, 4]$.