A. Birthday-Candies

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Today is Max's birthday and he is excited to meet his friends. But he is not equally fond of all of them and decided to distribute the candies accordingly. Also, some of his friends are very sweet and would give him some candies as a birthday present. Max being smart knows how many candies each of his friends would gift him. But he does not know the order in which he'll meet his friends. So he wants to calculate the minimum number of candies he'll have to buy, no matter what order they arrive in. He can distribute the candies he got as a gift from one friend to those who arrive after him or her but not to the same person as Max will be giving candies before receiving.

Input

The first line contains an integer n ($1 \leq n \leq 10^5$), which is the number of friends. Next n lines contain 2 numbers. The i^{th} line represents the i^{th} friend and it contains 2 values a_i($0 \leq a_i \leq 10^9$) and $b_i(0 \leq b_i \leq 10^9)$. a_i being the candies Max wants to gift him/her and b_i being the candies he'll recieve from the i^{th} friend.

Output

Minimum number of sweets he should buy. Note that the given input is not the order in which the friends arrive. You should output the minimum number of candies Max must buy so that whatever order his friends come in, he always has enough to give them.

Examples

input

```
5
12 3
9 0
7 6
5 4
0 2
```

output

```
26
```

input

```
3
0 2
5 6
7 0
```

output

```
12
```

Note

We'll have to find out the worst-case which for the above case is
12 3
9 0
5 4
7 6
0 2

For this we compute Max will need 12 for 1^{st} friend + 6 for 2^{nd} as he'll have 3 already + 5 for the 3^{rd} friend + 3 for the 4^{th} friend + 0 for 5^{th} Hence a total of 26 candies are required.

B. Birthday Gift

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Your little brother's birthday is approaching. You prepared an array of n positive integers $a_1, a_2, \ldots, a_n$ for him as a gift. But now you realized that he hates big numbers. So you decided to have the array changed. You have k coins to spare initially. You can select a number a_i of the array and do any of the following operations:

- Change a_i to $a_i - 1$. This costs c_1 coins.
- Change a_i to $[\frac{a_i}{2}]$. This costs c_2 coins. ($[.]$ denotes the greatest integer function.)

You can perform the above operations any number of times if you have the required number of coins. Also, it is allowed to select different elements of array in different operations. You would like to minimize the maximum element of the array. Find this minimum value.

Input

The first line contains 4 space separated integers n $(1 \le n \le 10^5)$, k $(1 \le k \le 10^9)$, c_1 $(1 \le c_1 \le 10^9)$ and c_2 $(1 \le c_2 \le 10^9)$ denoting the number of elements in the array, the number of coins you have, the cost of first operation and the cost of second operation respectively. The following line contains n space separated integers $a_1, a_2, \ldots a_n$ $(1 \le a_i \le 10^9)$ denoting the elements of the array.

Output

Print a single integer denoting the minimum possible value of the maximum element of the array that you can achieve.

Examples

input
5 20 3 5 4 6 12 11 8
output
6

input
4 15 1 5 8 10 12 14

output
7

Note

In the first example, we apply operation 2 on a_3, a_4, a_5. The new array is $[4, 6, 6, 5, 4]$ and we used $5 \times 3 = 15$ coins, with maximum element of array equal to 6. It can be verified that we cannot do better.

In the second example, we apply operation 1 one time on a_1 and three times on a_2. Then we apply operation 2 one time on both a_3 and a_4. Our final array is $[7, 7, 6, 7]$ with maximum of array equal to 7 and we used $1 \times 4 + 5 \times 2 = 14$ coins. It can be verified that we cannot do better.

C. AND OR XOR

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given 3 non-negative integers A, O and X. You need to find a multiset of non-negative integers such that their bitwise `AND` is A, their bitwise `OR` is O and their bitwise `XOR` is X. If there are many such multisets, you need to find one with minimum size.

The bitwise `AND`, `OR` and `XOR` of a multiset is the `AND`, `OR` and `XOR` of all its elements respectively. In case the multiset has a single element, the `AND`, `OR` and `XOR` are defined as the element itself. For eg.

1. For the multiset $[1, 1, 4]$ bitwise `OR`=$1|1|4 = 5$, bitwise `AND`= $1\&1\&4 = 0$, bitwise `XOR`=$1 \oplus 1 \oplus 4 = 4$.
2. For the multiset $[7]$ the bitwise `AND`, `OR` and `XOR` are all 7.

Input

The first line contains a single integer T $(1 \le T \le 10^5)$- the number of testcases. T lines follow. Each line contains 3 space separated integers- A, O, X $(1 \le A, O, X \le 10^{17})$.

Output

The answer to each testcase starts in a new line. If there is no such multiset, print -1. Otherwise, first print an integer X denoting the size of the multiset found. In the next line, print X space separated non-negative integers, denoting the elements of the multiset with bitwise `AND` equal to A, bitwise `OR` equal to O and bitwise `XOR` equal to X. If there are many possible multisets with minimum possible size, you may print any one of them.

Example

input
2 4 4 4 4 4 0

output
1 4 2 4 4

D. Worthy Winner

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

There are two participants in a Mathematics Olympiad. Their ages a_1 and a_2 and their roll numbers r_1 and r_2 are given. Cash prize N is to be given to the winner. A winner is worthy if $N - a_i$ is divisible by r_i where a_i is the age of winner and r_i is the roll number of winner. Count the number of possible values of $0 \leq N \leq k$ such that each participant can be a worthy winner.

Input

The first line contains a single integer T denoting the number of test cases. Then T lines follow. The only line of each test case contains five space separated integers k, r_1, r_2, a_1 and a_2 .

$1 \leq T \leq 10$

$0 \leq k \leq 10^9$

$1 \leq r_1, r_2 \leq 1000$

$0 \leq a_1 < r_1, 0 \leq a_2 < r_2$

Output

For each test case, output the number of possible values of $0 \leq N \leq k$.

Example

input
3 100 4 6 2 4 40 4 5 3 4 10 4 5 3 4

output
8 2 0

E. n-nice numbers

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

A pair of positive integers is called n-nice if the following holds:

1. The lengths of their binary representation is same.
2. The number of positions where the bit is different in their respective binary representations is exactly equal to 2^n.
3. The two numbers differ exactly by 2^n.

You are given an integer n $(1 \leq n \leq 20)$. Construct (if possible), an n-nice pair of positive integers whose binary representations have length at most 2×10^5

NOTE: The binary representations should start with '1'.

Input

The input contains only a single integer n ($1 \leq n \leq 20$).

Output

Print a single line containing two space separated binary strings of n-nice numbers. If it is not possible then print "`Impossible`" (case sensitive) without quotes. If there are multiple possible pairs, you may print any of them

Examples

input

```
1
```

output

```
1101 1011
```

input

```
2
```

output

```
100100011 100011111
```

Note

In the second test case, $n = 2$, so number of mismatches = $2^2 = 4$. Also the difference should be 4. You can verify that the given numbers satisfy the conditions.

F. Are you winning?

time limit per test: 1 second
memory limit per test: 256 megabytes

input: standard input

output: standard output

On a late Saturday evening, Bob and his Dad are playing a game. Bob's Dad chose k numbers (at random) from the first n numbers and fixed a permutation of those k numbers (at random) without telling Bob. Now, Bob has to guess the sequence. For Bob's first random guess, Bob's Dad coloured each element of the sequence with:

G (green) if the position and the number both are correct

Y (yellow) if the number is present in the chosen permutation, but the position is incorrect

R (red) if the number is not present in the permutation.

Just making the first guess exhausted Bob, and so his Dad will let him win if Bob can tell him the maximum probability (given the output of the first guess) of guessing the sequence correctly in the next attempt. Can you help Bob answer his Dad and sleep early?

Input

Two space-separated integers n, k $(1 \leq n \leq 10^5, 1 \leq k \leq n)$.

The next line contains k space-separated integers a_i $(1 \leq a_i \leq n, 1 \leq i \leq k, a_i \neq a_j$ for $i \neq j)$, the initial guess made by Bob.

The third line of the input contains a string s of length k, the marking done by Bob's Dad for Bob's initial guess (s consists of only G, Y, R characters).

Output

A single integer that equals the maximum possible probability of guessing the sequence correctly in the next attempt, $P.Q^{-1}$ modulo 10^9+7, where P and Q are coprime.

Example

input
7 4 2 3 1 4 YRGY
output
111111112

Note

The permutation 4512 has the best probability equal to $\frac{1}{9}$

G. Painting Tiles

time limit per test: 1 second

memory limit per test: 256 megabytes

input: standard input

output: standard output

There are a series of n tiles that must be painted with colours. The tiles of the same colour should be adjacent to each other, sharing an edge. The number of tiles of the same colour should be a power of 2, i.e., 2^i, where $i \geq 0$. Find the minimum number of colours required to paint all the tiles in this fashion.

Input

First line contains the number of test cases t ($1 \leq t \leq 1.5 \times 10^5$)

The only line of each test case contains a single integer n $(1 \leq n \leq 10^{18})$.

Output

For each test case, output a single integer, the minimum number of colours required.

Example

input
2 4 6
output
1 2

Note

For $n = 4$, all the tiles can be painted with the same colour.

For $n = 6$, paint 2 tiles with one colour and next 4 tiles with another.

H. Good Sequence

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

A good sequence of length n is defined as a sequence of integers $a_1, a_2, \ldots a_n$ which satisfies the following three conditions:-

1. $1 \leq a_i \leq 2^{32} - 1$, for all $1 \leq i \leq n$.
2. The binary representation of each a_i contains exactly two $1s$.
3. In the binary representations of all adjacent pairs of numbers, the distance between their left $1s$ should be atmost m, where m is an integer. The same applies to their right $1s$ (see the **notes** given below for a better explanation).

Input

The first line of the input consists of a single integer t ($1 \leq t \leq 100$), denoting the number of test cases.

Next t lines contain two integers n ($2 \leq n \leq 100$) and m ($0 \leq m \leq 10$) separated by a white space.

The sum of values of n across all the test cases does not exceed 100.

Output

The output consists of t lines where each line is an integer denoting the number of good sequences of length n that can be made using the given value of m. As the number might be large, output the no. of ways modulo $10^9 + 7$.

Example

input

```
1
2 0
```

output

```
496
```

Note

In the example case, $n = 2$ and $m = 0$. This means that we have to find two numbers, say a and b, where the left and right 1 in the binary representation of a are present at indices l_a and r_a and similarly, the the left and right indices of 1 in the binary representation of b are at l_b and r_b, such that $|l_a - l_b| \leq 0$ and $|r_a - r_b| \leq 0$. This implies that $l_a = l_b$ and $r_a = r_b$. Therefore the number of good sequences that can be made are $\binom{32}{2}$ = 496.

I. Pamdalchemist

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Pamda, the legendary alchemist dreamt of a legendary potion the previous night.

He knows the potion can be made using equal parts of exactly k chemicals of the available n chemicals, in a particular order. He knows that using a chemical of index j before a chemical of index i $(i < j)$ can cause a catastrophe, so that is forbidden.

He knows that the volatility of the i^{th} chemical is a_i. Also, by experience, he knows that we can only add chemicals with strictly higher volatility than all other chemicals already in the concoction and we can add only one chemical at a time.

Pamda wishes to know the number of correct ways to make the potion. As he is not-so-legendary at CP, please help him out.

Since the answer could be too large, print it modulo $10^9 + 7$.

Input

The first line contains two integer values n and k $(1 \leq n \leq 10^5, 1 \leq k \leq 11)$ — the total number of chemicals available and the number of chemicals required to make the potion.

The next line contains n integers, the i^{th} integer, a_i $(1 \leq a_i \leq 10^9)$ is the volatility of the chemical with index i.

Output

Print a single integer ans, equal to the number of valid ways to make the potion, modulo $10^9 + 7$.

Examples

input

```
6 3
1 1 3 2 2 3
```

output

```
4
```

input

```
6 2
1 2 3 4 5 6
```

output

```
15
```

J. Assignment Conundrum (Easy)

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The only difference between the easy and hard version is the constraint on k.

You are given n assignments and non-negative integer k. The i-th assignment is given to you on the i-th day and takes a_i days to complete. You can choose to do a subset of the assignments following some constraints:

1. You must start the assignment on the day it was given, that is if you decide to do the i-th assignment it must be started on the i-th day.
2. You can do at most one assignment in a single day.
3. Once you start an assignment you must continue doing it for the following a_i days, that is if you start assignment i you must spend the days $i, i+1, \ldots, i+a_i-1$ doing the i-th assignment. Note that $i+a_i-1$ may exceed n.
4. In the first n days, you can spend at most k consecutive days without doing an assignment, that is if $1 \leq i \leq n-k$ and you spend the days $i, i+1, \ldots, i+k-1$ without doing an assignment, then you must start assignment $i+k$.

Minimize the total number of days spent doing assignments.

Input

The first line contains two space-separated integers n $(1 \leq n \leq 2 \times 10^5)$ and k

$(0 \leq k \leq \min(n, 10))$.

The second line contains n space-separated integers representing the values $a_1, a_2, \ldots, a_n$ satisfying $1 \leq a_i \leq n$ for all $1 \leq i \leq n$.

Output

Print a single integer representing the minimum number of days spent doing assignments.

Examples

input

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

output

```
2
```

input

```
5 0
1 2 3 4 5
```

output

```
7
```

input

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

output

```
0
```

input

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

output

```
4
```

Note

1. We don't do any assignment on day 1, start assignment 2, this takes the days 2 and 3, then we don't do any assignments on day 4 and 5.
2. We do assignments 1, 2 and 4.
3. We skip all the assignments.
4. We skip assignment 1 and do 2.

K. Assignment Conundrum (Hard)

time limit per test: 3 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

The only difference between the easy and hard version is the constraint on k.

You are given n assignments and non-negative integer k. The i-th assignment is given to you on the i-th day and takes a_i days to complete. You can choose to do a subset of the assignments following some constraints:

1. You must start the assignment on the day it was given, that is if you decide to do the i-th assignment it must be started on the i-th day.
2. You can do at most one assignment in a single day.
3. Once you start an assignment you must continue doing it for the following a_i days, that is if you start assignment i you must spend the days $i, i+1, \ldots, i+a_i-1$ doing the i-th assignment. Note that $i+a_i-1$ may exceed n.
4. In the first n days, you can spend at most k consecutive days without doing an assignment, that is if $1 \le i \le n-k$ and you spend the days $i, i+1, \ldots, i+k-1$ without doing an assignment, then you must start assignment $i+k$.

Minimize the total number of days spent doing assignments.

Input

The first line contains two space-separated integers n $(1 \le n \le 2 \times 10^5)$ and k $(0 \le k \le n)$.

The second line contains n space-separated integers representing the values $a_1, a_2, \ldots, a_n$ satisfying $1 \le a_i \le n$ for all $1 \le i \le n$.

Output

Print a single integer representing the minimum number of days spent doing assignments.

Examples

input
5 2 1 2 3 4 5
output
2

input
5 0 1 2 3 4 5
output
7

input
5 5 1 2 3 4 5
output
0

input
5 1 5 4 3 2 1
output
4

Note

1. We don't do any assignment on day 1, start assignment 2, this takes the days 2 and 3, then we don't do any assignments on day 4 and 5.
2. We do assignments 1, 2 and 4.
3. We skip all the assignments.
4. We skip assignment 1 and do 2.

L. Square it up!

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Jack loves playing with numbers so he asked his mathematics teacher to give him a challenging question. His teacher knew Jack was a brilliant student and so he immediately thought of a question to test his student's ability.

He gave Jack two positive integers n and m, and then asked Jack to find a positive integer k, and k **positive** integers $x_1, x_2, \ldots, x_k$ such that

1. k is as small as possible.
2. The sum of the k integers is n, that is $\sum_{i=1}^{k} x_i = n$.
3. The sum of squares of the k integers is m, that is $\sum_{i=1}^{k} x_i^2 = m$.

For example, if $n = 12$, $m = 42$ then $x = [1, 3, 4, 4], [2, 2, 3, 5], [2, 3, 5, 2]$ are all valid solutions whereas $x = [1, 1, 1, 1, 1, 1, 6]$ is not.

For $n = 28$, $m = 420$, $x = [1, 1, 3, 3, 20]$ is a valid solution but $x = [-1, 14, 15]$ is not.

Input

Each test contains multiple test cases. The first line contains the number of test cases t $(1 \leq t \leq 10^5)$.

The next t lines contains two space-separated integers n $(1 \leq n \leq 100)$ and m $(1 \leq m \leq 10^9)$.

The input is chosen such that the sum of k over all testcases is less than 10^5.

Output

For each test case, if no solution exists then print a single line containing -1.

Otherwise print two lines, the first line containing k followed by k space separated integers, $x_1, x_2, \ldots, x_k$ on the next line.

If there are multiple possible answers, print any.

Example

input

```
6
12 42
12 42
12 42
2 11
28 420
16 10000000
```

output

```
4
1 3 4 4
4
2 2 3 5
4
2 3 5 2
-1
5
1 1 3 3 20
-1
```

M. Awesome Strings

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Bob is given a string of length n containing uppercase English characters. He wants to find out the minimum number of operations needed to make the string awesome. An awesome string is one in which the the strings formed from the characters on the even and odd indices are non increasing and non decreasing respectively.

In other words an awesome string obeys the following criterion for all j ($1 \leq j \leq n-2$):

1. If j is odd, the character a_j is equal to a_{j+2} or comes before a_{j+2} in the English alphabet.

2. if j is even, the character a_j is equal to a_{j+2} or comes after a_{j+2} in the English alphabet

In every operation Bob can replace any character of the string with any other uppercase English character of his choice.

Input

The first line contains the length of the string n ($1 \leq n \leq 10^5$).

The second line contains the string of length n, consisting of uppercase English characters.

Output

Output the minimum number of operations that can make the string awesome.

Examples

input
5 RAPMQ
output
2

input
4 ADCA
output
0

Note

In the first case, RPQ and AM are the odd and even substrings respectively. To make the odd substring non decreasing, replace R by O. To make the even substring non increasing, replace A by N. Hence minimum number of operations to make the string awesome is 2.

In the second case, the string is already awesome. Hence no operations are required.

N. XYZ wants maximum money

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

XYZ is given an array of n coins where each element of the array represents the value of a coin. The coins are not necessarily distinct and he has to select coins such that all the coins have the same value. He wants to know the maximum amount of money he can obtain and the coin value to be selected to obtain this amount. He asks for your help to do this.

Please note: If the same maximum amount of money can be made by coins of different values, then pick the coins with greater value.

Input

The first line of the input consists of a integer t ($1 \le t \le 40$), denoting the number of test cases. The next $2t$ lines contain each test case.

Each testcase consists of two lines. The first line contains the integer n ($1 \le n \le 10^5$), representing the total number of coins in the array.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$), where a_i represents the value of the ith coin.

The sum of n over all test cases is $\leq 10^5$.

Output

The output should contain t lines where each line consists of two integers separated by a space, where the first integer represents the value of the coin that is to be selected and the second integer represents the maximum amount of money that can be obtained.

Example

input

```
2
7
2 2 3 2 1 1 2
4
1 1 1 4
```

output

```
2 8
4 4
```