Codeforces Round #631 (Div. 1) - Thanks, Denis aramis Shitov!

A. Dreamoon Likes Coloring

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

Dreamoon likes coloring cells very much.

There is a row of n cells. Initially, all cells are empty (don't contain any color). Cells are numbered from 1 to n.

You are given an integer m and m integers $l_1, l_2, \ldots, l_m$ ($1 \le l_i \le n$)

Dreamoon will perform m operations.

In i-th operation, Dreamoon will choose a number p_i from range $[1, n-l_i+1]$ (inclusive) and will paint all cells from p_i to p_i+l_i-1 (inclusive) in i-th color. Note that cells may be colored more one than once, in this case, cell will have the color from the latest operation.

Dreamoon hopes that after these m operations, all colors will appear at least once and all cells will be colored. Please help Dreamoon to choose p_i in each operation to satisfy all constraints.

Input

The first line contains two integers n, m ($1 \le m \le n \le 100\,000$).

The second line contains m integers $l_1, l_2, \ldots, l_m$ ($1 \le l_i \le n$).

Output

If it's impossible to perform m operations to satisfy all constraints, print "'-1" (without quotes).

Otherwise, print m integers $p_1, p_2, \ldots, p_m$ ($1 \le p_i \le n - l_i + 1$), after these m operations, all colors should appear at least once and all cells should be colored.

If there are several possible solutions, you can print any.

Examples

input

```
5 3
3 2 2
```

output

```
2 4 1
```

input

```
10 1
1
```

output

```
-1
```

B. Dreamoon Likes Sequences

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

Dreamoon likes sequences very much. So he created a problem about the sequence that you can't find in OEIS:

You are given two integers d, m, find the number of arrays a, satisfying the following constraints:

- The length of a is n, $n \ge 1$
- $1 \le a_1 < a_2 < \cdots < a_n \le d$
- Define an array b of length n as follows: $b_1 = a_1$, $\forall i > 1, b_i = b_{i-1} \oplus a_i$, where $\oplus$ is the bitwise exclusive-or (xor). After constructing an array b, the constraint $b_1 < b_2 < \cdots < b_{n-1} < b_n$ should hold.

Since the number of possible arrays may be too large, you need to find the answer modulo m.

Input

The first line contains an integer t ($1 \le t \le 100$) denoting the number of test cases in the input.

Each of the next t lines contains two integers d, m ($1 \le d, m \le 10^9$).

Note that m is not necessary the prime!

Output

For each test case, print the number of arrays a, satisfying all given constrains, modulo m.

Example

input

```
10
1 1000000000
2 999999999
3 99999998
4 9999997
5 999996
6 99995
7 9994
8 993
9 92
10 1
```

output

```
1
3
5
11
17
23
29
59
89
0
```

C. Drazil Likes Heap

time limit per test: 1.5 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Drazil likes heap very much. So he created a problem with heap:

There is a max heap with a height h implemented on the array. The details of this heap are the following:

This heap contains exactly $2^h - 1$ **distinct** positive non-zero integers. All integers are distinct. These numbers are stored in the array a indexed from 1 to $2^h - 1$. For any $1 < i < 2^h$, $a[i] < a[\left\lfloor \frac{i}{2} \right\rfloor]$.

Now we want to reduce the height of this heap such that the height becomes g with exactly $2^g - 1$ numbers in heap. To reduce the height, we should perform the following action $2^h - 2^g$ times:

Choose an index i, which contains an element and call the following function f in index i:

Algorithm 1 The function f

```
procedure $f(i)$
    left_node_id $\leftarrow 2i$
    right_node_id $\leftarrow 2i + 1$
    if $a$[left_node_id] $= 0$ and $a$[right_node_id] $= 0$ then
        $a[i] \leftarrow 0$
    else
        if $a$[left_node_id] $>$ $a$[right_node_id] then
            $a[i] \leftarrow a$[left_node_id]
            $f$(left_node_id)
        else
            $a[i] \leftarrow a$[right_node_id]
            $f$(right_node_id)
        end if
    end if
end procedure
```

Note that we suppose that if $a[i] = 0$, then index i don't contain an element.

After all operations, the remaining $2^g - 1$ element must be located in indices from 1 to $2^g - 1$. Now Drazil wonders what's the minimum possible sum of the remaining $2^g - 1$ elements. Please find this sum and find a sequence of the function calls to achieve this value.

Input

The first line of the input contains an integer t ($1 \le t \le 70\,000$): the number of test cases.

Each test case contain two lines. The first line contains two integers h and g ($1 \le g < h \le 20$). The second line contains $n = 2^h - 1$ **distinct** positive integers $a[1], a[2], \ldots, a[n]$ ($1 \le a[i] < 2^{20}$). For all i from 2 to $2^h - 1$, $a[i] < a[\left\lfloor \frac{i}{2} \right\rfloor]$.

The total sum of n is less than 2^{20}.

Output

For each test case, print two lines.

The first line should contain one integer denoting the minimum sum after reducing the height of heap to g. The second line should contain $2^h - 2^g$ integers $v_1, v_2, \ldots, v_{2^h-2^g}$. In i-th operation $f(v_i)$ should be called.

Example

input

```
2
3 2
7 6 3 5 4 2 1
3 2
7 6 5 4 3 2 1
```

output

```
10
3 2 3 1
8
2 1 3 1
```

D. Dreamoon Likes Strings

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

Dreamoon likes strings. Today he created a game about strings:

String $s_1, s_2, \ldots, s_n$ is **beautiful** if and only if for each $1 \le i < n, s_i \ne s_{i+1}$.

Initially, Dreamoon has a string a. In each step Dreamoon can choose a **beautiful** substring of a and remove it. Then he should concatenate the remaining characters (in the same order).

Dreamoon wants to use the smallest number of steps to make a empty. Please help Dreamoon, and print any sequence of the smallest number of steps to make a empty.

Input

The first line contains an integer t ($1 \le t \le 200\,000$), denoting the number of test cases in the input.

For each test case, there's one line with a non-empty string of lowercase Latin letters a.

The total sum of lengths of strings in all test cases is at most $200\,000$.

Output

For each test case, in the first line, you should print m: the smallest number of steps to make a empty. Each of the following m lines should contain two integers l_i, r_i ($1 \le l_i \le r_i \le |a|$), denoting, that the i-th step is removing the characters from index l_i to r_i in the current string. (indices are numbered starting from 1).

Note that after the deletion of the substring, indices of remaining characters may change, and r_i should be at most the current length of a.

If there are several possible solutions, you can print any.

Example

input

```
4
aabbcc
aaabbb
aaa
abacad
```

output

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

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

E. Dreamoon Loves AA

time limit per test: 3.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

There is a string of length $n+1$ of characters 'A' and 'B'. The first character and last character of the string are equal to 'A'.

You are given m indices $p_1, p_2, \ldots, p_m$ (0-indexation) denoting the other indices of characters 'A' in the string.

Let's denote the minimum distance between two neighboring 'A' as l, and maximum distance between neighboring 'A' as r.

For example, (l, r) of string "ABBAABBBA" is $(1, 4)$.

And let's denote the **balance degree** of a string as the value of $r - l$.

Now Dreamoon wants to change exactly k characters from 'B' to 'A', and he wants to make the **balance degree** of the string as small as possible.

Please calculate the required minimum possible value of balance degree.

Input

The first line contains one integer t denoting the number of test cases ($1 \le t \le 400\,000$).

For each test case, the first line contains three integers n, m and k ($1 \le n \le 10^{15}, 0 \le m \le 400\,000, 0 \le k < n - m$).

The second line contains m integers $p_1, p_2, \ldots, p_m$, ($0 < p_1 < p_2 < \ldots < p_m < n$).

The total sum of m is at most $400\,000$.

Output

For each test case, print one integer: the smallest possible value of balance degree after k changes of 'B' to 'A'.

Example

input

```
5
80 3 5
11 24 50
81 7 12
4 10 17 26 37 48 61
25 10 14
3 4 7 12 13 15 17 19 21 23
1 0 0

10 2 0
2 4
```

output

```
5
2
0
0
4
```