Educational Codeforces Round 52 (Rated for Div. 2)

A. Vasya and Chocolate

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

There is a special offer in Vasya's favourite supermarket: if the customer buys a chocolate bars, he or she may take b additional bars for free. This special offer can be used any number of times.

Vasya currently has s roubles, and he wants to get as many chocolate bars for free. Each chocolate bar costs c roubles. Help Vasya to calculate the maximum possible number of chocolate bars he can get!

Input

The first line contains one integer t $(1 \le t \le 100)$ — the number of testcases.

Each of the next t lines contains four integers s, a, b, c $(1 \le s, a, b, c \le 10^9)$ — the number of roubles Vasya has, the number of chocolate bars you have to buy to use the special offer, the number of bars you get for free, and the cost of one bar, respectively.

Output

Print t lines. i-th line should contain the maximum possible number of chocolate bars Vasya can get in i-th test.

Example

input Copy

```
2
10 3 1 1
1000000000 1 1000000000 1
```

output

```
13
1000000001000000000
```

Note

In the first test of the example Vasya can buy 9 bars, get 3 for free, buy another bar, and so he will get 13 bars.

In the second test Vasya buys 1000000000 bars and gets 1000000000000000000 for free. So he has 1000000001000000000 bars.

B. Vasya and Isolated Vertices

time limit per test: 1 second

memory limit per test: 256 megabytes

input: standard input

output: standard output

Vasya has got an undirected graph consisting of n vertices and m edges. This graph doesn't contain any self-loops or multiple edges. Self-loop is an edge connecting a vertex to itself. Multiple edges are a pair of edges such that they connect the same pair of vertices. Since the graph is undirected, the pair of edges $(1, 2)$ and $(2, 1)$ is considered to be multiple edges. Isolated vertex of the graph is a vertex such that there is no edge connecting this vertex to any other vertex.

Vasya wants to know the minimum and maximum possible number of isolated vertices in an undirected graph consisting of n vertices and m edges.

Input

The only line contains two integers n and m $(1 \le n \le 10^5, 0 \le m \le \frac{n(n-1)}{2})$.

It is guaranteed that there exists a graph without any self-loops or multiple edges with such number of vertices and edges.

Output

In the only line print two numbers min and max — the minimum and maximum number of isolated vertices, respectively.

Examples

input

```
4 2
```

output
```
0 1
```

input
```
3 1
```

output
```
1 1
```

Note

In the first example it is possible to construct a graph with 0 isolated vertices: for example, it should contain edges $(1, 2)$ and $(3, 4)$. To get one isolated vertex, we may construct a graph with edges $(1, 2)$ and $(1, 3)$.

In the second example the graph will always contain exactly one isolated vertex.

C. Make It Equal

time limit per test: 2 seconds

memory limit per test: 256 megabytes

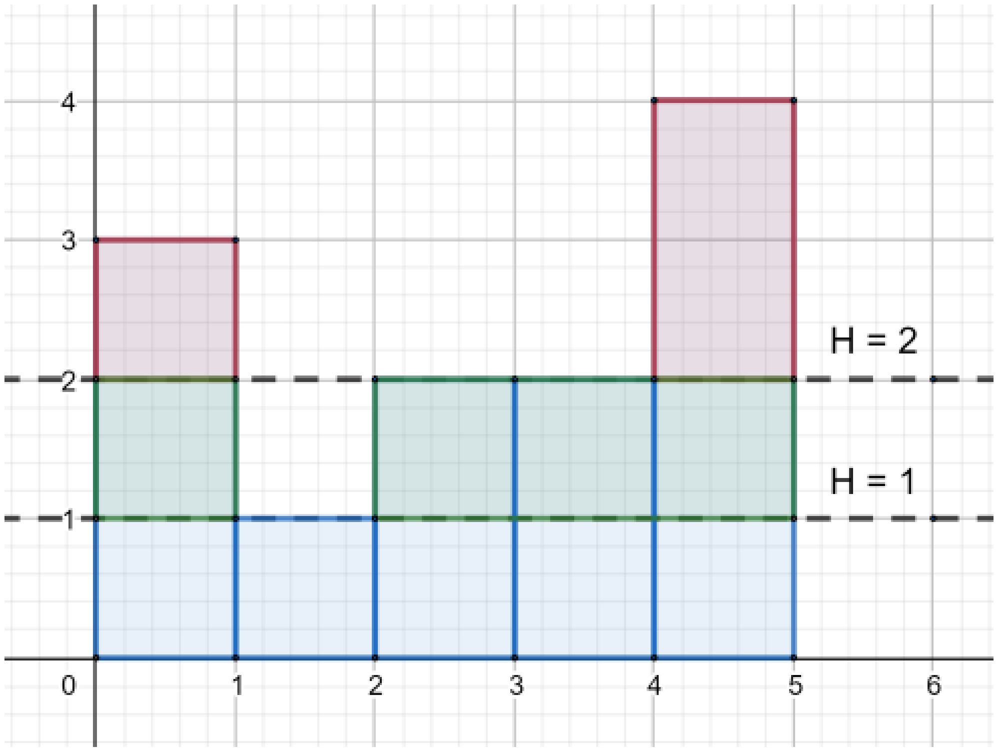
input: standard input

output: standard output

There is a toy building consisting of n towers. Each tower consists of several cubes standing on each other. The i-th tower consists of h_i cubes, so it has height h_i.

Let's define operation *slice* on some height H as following: for each tower i, if its height is greater than H, then remove some top cubes to make tower's height equal to H. Cost of one "slice" equals to the total number of removed cubes from all towers.

Let's name slice as *good* one if its cost is lower or equal to k ($k \ge n$).



Calculate the minimum number of good slices you have to do to make all towers have the same height. Of course, it is always possible to make it so.

Input

The first line contains two integers n and k ($1 \le n \le 2 \cdot 10^5$, $n \le k \le 10^9$) — the number of towers and the restriction on slices, respectively.

The second line contains n space separated integers $h_1, h_2, \dots, h_n$ ($1 \le h_i \le 2 \cdot 10^5$) — the initial heights of towers.

Output

Print one integer — the minimum number of good slices you have to do to make all towers have the same heigth.

Examples

input Copy

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

output Copy

```
2
```

input

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

output

```
2
```

Note

In the first example it's optimal to make 2 slices. The first slice is on height 2 (its cost is 3), and the second one is on height 1 (its cost is 4).

D. Three Pieces

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

You stumbled upon a new kind of chess puzzles. The chessboard you are given is not necesserily 8×8, but it still is $N \times N$. Each square has some number written on it, all the numbers are from 1 to N^2 and all the numbers are pairwise distinct. The j-th square in the i-th row has a number A_{ij} written on it.

In your chess set you have only three pieces: a knight, a bishop and a rook. At first, you put one of them on the square with the number 1 (you can choose which one). Then you want to reach square 2 (possibly passing through some other squares in process), then square 3 and so on until you reach square N^2. In one step you are allowed to either make a valid move with the current piece or replace it with some other piece. **Each square can be visited arbitrary number of times**.

A knight can move to a square that is two squares away horizontally and one square vertically, or two squares vertically and one square horizontally. A bishop moves diagonally. A rook moves horizontally or vertically. The move should be performed to a different square from the one a piece is currently standing on.

You want to minimize the number of steps of the whole traversal. Among all the paths to have the same number of steps you want to choose the one with the lowest number of piece replacements.

What is the path you should take to satisfy all conditions?

Input

The first line contains a single integer N $(3 \le N \le 10)$ — the size of the chessboard.

Each of the next N lines contains N integers $A_{i1}, A_{i2}, \dots, A_{iN}$ $(1 \le A_{ij} \le N^2)$ — the numbers written on the squares of the i-th row of the board.

It is guaranteed that all A_{ij} are pairwise distinct.

Output

The only line should contain two integers — the number of steps in the best answer and the number of replacement moves in it.

Example

input

```
3
1 9 3
8 6 7
4 2 5
```

output

```
12 1
```

Note

Here are the steps for the first example (the starting piece is a knight):

1. Move to $(3, 2)$
2. Move to $(1, 3)$
3. Move to $(3, 2)$
4. Replace the knight with a rook
5. Move to $(3, 1)$
6. Move to $(3, 3)$
7. Move to $(3, 2)$
8. Move to $(2, 2)$
9. Move to $(2, 3)$
10. Move to $(2, 1)$

11. Move to $(1, 1)$
12. Move to $(1, 2)$

E. Side Transmutations

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Consider some set of distinct characters A and some string S, consisting of exactly n characters, where each character is present in A.

You are given an array of m integers b ($b_1 < b_2 < \dots < b_m$).

You are allowed to perform the following move on the string S:

1. Choose some valid i and set $k = b_i$;
2. Take the first k characters of $S = Pr_k$;
3. Take the last k characters of $S = Su_k$;
4. Substitute the first k characters of S with the reversed Su_k;
5. Substitute the last k characters of S with the reversed Pr_k.

For example, let's take a look at $S =$ "`abcdefghi`" and $k = 2$. $Pr_2 =$ "`ab`", $Su_2 =$ "`hi`". Reversed $Pr_2 =$ "`ba`", $Su_2 =$ "`ih`". Thus, the resulting S is "**`ih`**`cdefg`**`ba`**".

The move can be performed arbitrary number of times (possibly zero). Any i can be selected multiple times over these moves.

Let's call some strings S and T equal if and only if there exists such a sequence of moves to transmute string S to string T. For the above example strings "`abcdefghi`" and "`ihcdefgba`" are equal. Also note that this implies $S = S$.

The task is simple. Count the number of distinct strings.

The answer can be huge enough, so calculate it modulo 998244353.

Input

The first line contains three integers n, m and $|A|$ ($2 \le n \le 10^9, 1 \le m \le min(\frac{n}{2}, 2 \cdot 10^5), 1 \le |A| \le 10^9$) — the length of the strings, the size of the array b and the size of the set A, respectively.

The second line contains m integers $b_1, b_2, \dots, b_m$ ($1 \le b_i \le \frac{n}{2}, b_1 < b_2 < \dots < b_m$).

Output

Print a single integer — the number of distinct strings of length n with characters from set A modulo 998244353.

Examples

input Copy

```
3 1 2
1
```

output Copy

```
6
```

input Copy

```
9 2 26
2 3
```

output Copy

```
150352234
```

input Copy

```
12 3 1
2 5 6
```

output Copy

```
1
```

Note

Here are all the distinct strings for the first example. The chosen letters 'a' and 'b' are there just to show that the characters in A are different.

1. "aaa"
2. "aab" = "baa"

3. "aba"
4. "abb" = "bba"
5. "bab"
6. "bbb"

F. Up and Down the Tree

time limit per test: 3 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

You are given a tree with n vertices; its root is vertex 1. Also there is a token, initially placed in the root. You can move the token to other vertices. Let's assume current vertex of token is v, then you make any of the following two possible moves:

- move down to any **leaf** in subtree of v;
- if vertex v is a leaf, then move up to the parent no more than k times. In other words, if $h(v)$ is the depth of vertex v (the depth of the root is 0), then you can move to vertex to such that to is an ancestor of v and $h(v) - k \le h(to)$.

Consider that root is not a leaf (even if its degree is 1). Calculate the maximum number of different leaves you can visit during one sequence of moves.

Input

The first line contains two integers n and k ($1 \le k < n \le 10^6$) — the number of vertices in the tree and the restriction on moving up, respectively.

The second line contains $n-1$ integers $p_2, p_3, \dots, p_n$, where p_i is the parent of vertex i.

It is guaranteed that the input represents a valid tree, rooted at 1.

Output

Print one integer — the maximum possible number of different leaves you can visit.

Examples

input

Copy

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

output

```
4
```

input

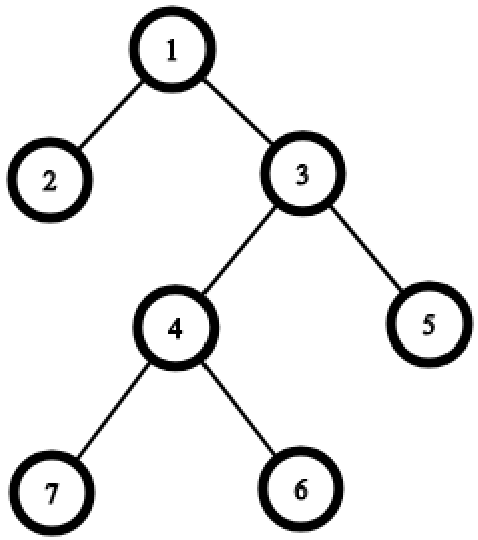
```
8 2
1 1 2 3 4 5 5
```

output

```
2
```

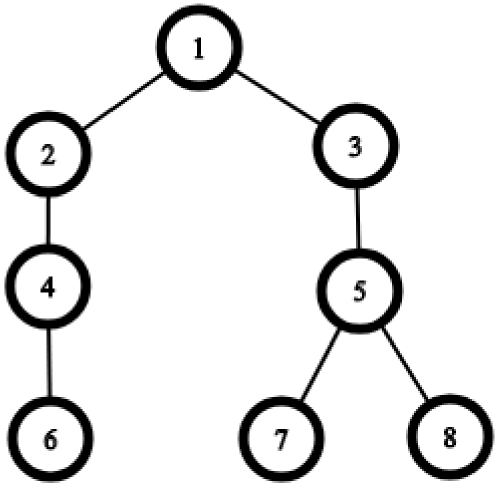
Note

The graph from the first example:



One of the optimal ways is the next one: $1 \to 2 \to 1 \to 5 \to 3 \to 7 \to 4 \to 6$.

The graph from the second example:



One of the optimal ways is the next one: $1 \to 7 \to 5 \to 8$. Note that there is no way to move from 6 to 7 or 8 and vice versa.

G. Fibonacci Suffix

time limit per test: 1 second

memory limit per test: 256 megabytes

input: standard input

output: standard output

Let's denote (yet again) the sequence of Fibonacci strings:

$F(0) = \texttt{0}$, $F(1) = \texttt{1}$, $F(i) = F(i-2) + F(i-1)$, where the plus sign denotes the concatenation of two strings.

Let's denote the **lexicographically sorted** sequence of suffixes of string $F(i)$ as $A(F(i))$. For example, $F(4)$ is `01101`, and $A(F(4))$ is the following sequence: `01`, `01101`, `1`, `101`, `1101`. Elements in this sequence are numbered from 1.

Your task is to print m first characters of k-th element of $A(F(n))$. If there are less than m characters in this suffix, then output the whole suffix.

Input

The only line of the input contains three numbers n, k and m ($1 \leq n, m \leq 200, 1 \leq k \leq 10^{18}$) denoting the index of the Fibonacci string you have to consider, the index of the element of $A(F(n))$ and the number of characters you have to output, respectively.

It is guaranteed that k does not exceed the length of $F(n)$.

Output

Output m first characters of k-th element of $A(F(n))$, or the whole element if its length is less than m.

Examples

input

```
4 5 3
```

output

```
110
```

input

```
4 3 3
```

output

```
1
```