Epiphany 10.1

A. Total Path

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

You are standing at point $(0, 0)$ in the 2-dimensional plane. Find the total number of ways to reach point (X, Y) using the minimum number of moves.

As it can be very large, so print it under modulo $10^9 + 7$.

If you are at the point (i, j) in one move you can move to any one adjacent point out of 4.

- $(i - 1, j)$: moving left
- $(i + 1, j)$: moving right
- $(i, j - 1)$: moving down
- $(i, j - 1)$: moving up

Input

The input consists of multiple test cases. The first line contains an integer t ($1 \le t \le 10^5$) — the number of test cases. The description of the test cases follows.

The first and only line of each test case contains two integers X ($-10^6 \le N \le 10^6$) and Y ($-10^6 \le M \le 10^6$)

Output

For each test case, Print one integer denoting the total number of ways to reach (X, Y) under modulo $10^9 + 7$

Example

input Copy

```
4
3 2
0 0
-4 -3
-1 2
```

output Copy

```
10
1
35
3
```

B. Defend the Walls

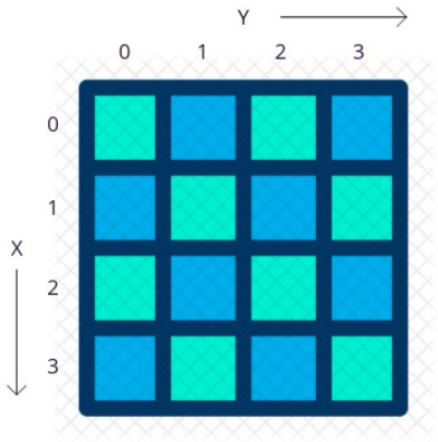
time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are Levi Ackerman.

In this world where humanity has survived for over a century, within the protection of the walls over 50 metres tall, one day, a gigantic titan peeks over these walls.

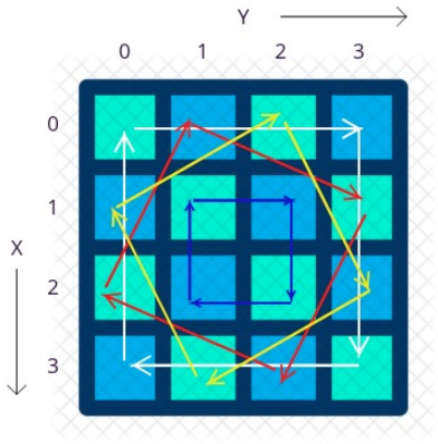
Being a Squad Captain, you are asked to defend the Walls from these titans using only your wit and the device given to you, famously known as the 3-D Manoeuvring Gear.

The Walls are defined as a 2D Square Matrix given in the Image below (for $N = 4$):



Using your 3-D Manoeuvring Gear, the path that you can follow is given in the below diagram (for $N = 4$):

Using your 3-D Maneuvring Gear, the path that you can follow is given in the below diagram (for $N = 4$).



For each coordinate (i, j) that lies at the tail of the arrow, you can reach only to the head of the arrow, as shown in the diagram.

With each Jump you take with the help of your Gear, you reach to the next position from your current position.

Find your Position after K such jumps, given the values of N, X, Y, K where:

N : The dimension of the 2D Square Matrix

X : X coordinate of your Current Position

Y : Y coordinate of your Current Position

K : Number of jumps you take from your Current Position

Input

The input consists of multiple test cases. The first line contains an integer t ($1 \leq t \leq 10^5$) — the number of test cases. The description of the test cases follows.

The first and only line of each test case contains four integers:

N ($1 \leq N \leq 10^{18}$)

X ($0 \leq X < N$)

Y ($0 \leq Y < N$)

K ($0 \leq K \leq 10^{18}$)

Output

For each test case, Print two integers in a single line denoting your final position.

Example

input Copy

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

output Copy

```
1 3
3 2
3 0
2 2
2 3
```

Note

The Output is self explanatory with the help of the diagram given in the Problem Statement.

C. Max Xor

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

Sharon and Steve are playing a game. There is an array arr of N numbers.

Sharon will ask Q question. In each question, she will give two numbers X and M to steve, and steve has to find an array value such that $arr[i] \leq M$ and $X \oplus arr[i]$ is maximum.($1 \leq i \leq N$)

If no such element is present print -1

Input

The input consists of multiple test cases. The first line contains an integer t ($1 \leq t \leq 5$) — the number of test cases. The description of the test cases follows.

The First line of each test case contains an integer N ($1 \leq N \leq 10^5$).

The Second line of each test case contains N space-separated values denoting $arr[i]$.($0 \leq arr[i] \leq 10^9$).

The third line of each test case contains an integer Q ($1 \leq Q \leq 10^4$) denoting the number of queries.

Next Q lines contains two integer X($0 \leq X \leq 10^9$) and M ($0 \leq M \leq 10^9$).

Output

For each test case,

Print Q integers denoting the answer to each query separated by a line.

Example

input Copy

```
2
10
67 72 49 42 88 62 35 65 60 53
5
31 21
73 65
96 1
75 25
64 49
10
31 67 55 86 72 17 69 30 90 97
5
57 67
84 27
3 23
87 5
92 99
```

output Copy

```
-1
53
-1
-1
49
67
17
17
-1
55
```

D. Maggie in the city

time limit per test: 1 s.
memory limit per test: 256 MB
input: standard input
output: standard output

Maggie got a job as an architect. Her first task was to connect n cities with the help of m two-way direct road so that it becomes possible to travel from one city to any other city via these roads. Each direct road has to link two different cities, each pair of cities should have at most one direct road. Evil enemy of mankind named SevUsal, made maggie an offer that she couldn't refuse: Maggie was asked to connect the cities in such a way, that when the city with index v cuts off its incoming and outgoing roads(no one can enter or leave this city), it becomes impossible to travel between some other two cities, i.e. the cities stops being connected. Help Maggie connect the cities.

Input

The first input line contains 3 space-separated integer numbers n, m, v ($3 \leq n \leq 10^5, 0 \leq m \leq 10^5, 1 \leq v \leq n$), n — number of cities, m — number of roads to be made, v — index of the city whose cutoff should lead to the failure of the whole city system.

Output

If it is impossible to connect the cities in the required way, output -1. Otherwise output m lines with 2 numbers each — description of all the direct roads. Each direct road is described by two numbers — indexes of two cities, linked by this direct road. The cities are numbered from 1. If the answer is not unique, output any.

Examples

input Copy

```
5 6 3
```

output Copy

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

input Copy

```
6 100 1
```

output Copy

```
-1
```

E. Tree Under Control

time limit per test: 2 s.
memory limit per test: 256 MB
input: standard input

output: standard output

Mike is given an undirected weighted tree, in which 1 is the root vertex. Control value of each vertex denoted by $c_1, c_2, \ldots, c_n$ (where n is the total number of vertices) is given to him. Let's take two vertices (V, U) such that V is the ancestor of U, and V controls U if the distance between U and V is less than or equal to the control value of U.

Note that here distance means the sum of the weights of the edges in the simple path between V to U.

Mike is asked to find the number of vertices control by each vertex. Can you help him.

Input
The first line contains single integer n ($1 \leq n \leq 2 \cdot 10^5$).

The second line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \leq c_i \leq 10^9$) — the control value of i-th vertices.

The next $(n-1)$ lines contain two integers each. The i-th of these lines contains integers p_i and w_i ($1 \leq p_i \leq n$, $1 \leq w_i \leq 10^9$) — the parent of the $(i+1)$-th vertex in the tree and the weight of the edge between p_i and $(i+1)$.

Output
Print n integers — the i-th of these numbers should be equal to the number of vertices that the i-th vertex controls.

Examples

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

output Copy
```
3 2 1 0 0 0
```

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

output Copy
```
4 2 0 0 0
```

F. Mike's Array

time limit per test: 2 s.
memory limit per test: 256 MB
input: standard input
output: standard output

Mike has an array that consists of n integers: $arr_1, arr_2, \cdots, arr_n$.

The hardness of this array arr is calculated by the formula:

$$fun(arr) = \max_{1 \leq i \leq n-1} |arr_{i+1} - arr_i|, n > 1; fun(arr) = 0, 0 \leq n \leq 1.$$

Mike wants to make this array as easier as possible by changing the values of at most k array elements by any integers.

Help Mike and calculate what minimum number $fun(arr)$ he can reach.

Input
The first line contains two integers n and k ($1 \leq k \leq n \leq 2000$). The second line contains space-separated integers $arr_1, arr_2, \cdots, arr_n$ ($-10^9 \leq arr_i \leq 10^9$).

Output
A single number — the minimum value of $fun(arr)$ Mike can get.

Examples

input Copy
```
5 2
4 7 4 7 4
```

output Copy
```
0
```

input Copy
```
3 1
-100 0 100
```

output Copy
```
100
```

input Copy
```
6 3
1 2 3 7 8 9
```

output Copy

```
1
```

Note

In the first sample, Mike can change the second and fourth elements and get array: $4, 4, 4, 4, 4$.

In the third sample he can get array: $1, 2, 3, 4, 5, 6$.