Codeforces Round 869 (Div. 1)

A. Almost Increasing Subsequence

2 seconds, 256 megabytes

A sequence is *almost-increasing* if it does not contain three **consecutive** elements x, y, z such that $x \geq y \geq z$.

You are given an array $a_1, a_2, \ldots, a_n$ and q queries.

Each query consists of two integers $1 \leq l \leq r \leq n$. For each query, find the length of the longest *almost-increasing* subsequence of the subarray $a_l, a_{l+1}, \ldots, a_r$.

A subsequence is a sequence that can be derived from the given sequence by deleting zero or more elements without changing the order of the remaining elements.

Input

The first line of input contains two integers, n and q ($1 \leq n, q \leq 200\,000$) — the length of the array a and the number of queries.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^9$) — the values of the array a.

Each of the next q lines contains the description of a query. Each line contains two integers l and r ($1 \leq l \leq r \leq n$) — the query is about the subarray $a_l, a_{l+1}, \ldots, a_r$.

Output

For each of the q queries, print a line containing the length of the longest almost-increasing subsequence of the subarray $a_l, a_{l+1}, \ldots, a_r$.

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

output
3 4 3 1 4 2 7 1

In the first query, the subarray is $a_1, a_2, a_3 = [1, 2, 4]$. The whole subarray is almost-increasing, so the answer is 3.

In the second query, the subarray is $a_1, a_2, a_3, a_4 = [1, 2, 4, 3]$. The whole subarray is a almost-increasing, because there are no three consecutive elements such that $x \geq y \geq z$. So the answer is 4.

In the third query, the subarray is $a_2, a_3, a_4, a_5 = [2, 4, 3, 3]$. The whole subarray is not almost-increasing, because the last three elements satisfy $4 \geq 3 \geq 3$. An almost-increasing subsequence of length 3 can be found (for example taking $a_2, a_3, a_5 = [2, 4, 3]$). So the answer is 3.

B. Fish Graph

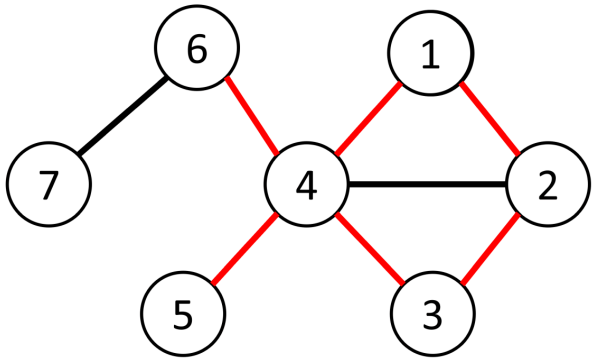
1 second, 256 megabytes

You are given a simple undirected graph with n nodes and m edges. Note that the graph is not necessarily connected. The nodes are labeled from 1 to n.

We define a graph to be a *Fish Graph* if it contains a simple cycle with a special node u belonging to the cycle. Apart from the edges in the cycle, the graph should have exactly 2 extra edges. Both edges should connect to node u, but they should not be connected to any other node of the cycle.

Determine if the graph contains a subgraph that is a Fish Graph, and if so, find any such subgraph.

In this problem, we define a subgraph as a graph obtained by taking any subset of the edges of the original graph.



Visualization of example 1. The red edges form one possible subgraph that is a Fish Graph.

Input

The first line of input contains the integer t ($1 \leq t \leq 1000$), the number of test cases. The description of test cases follows.

The first line of each test case contains two integers, n and m ($1 \leq n, m \leq 2\,000$) — the number of nodes and the number of edges.

Each of the next m lines contains the description of an edge. Each line contains two integers u_i and v_i ($1 \leq u_i, v_i \leq n$, $u_i \neq v_i$) — an edge connects node u_i to node v_i.

It is guaranteed that no two edges connect the same unordered pair of nodes.

Furthermore, it is guaranteed that the sum of n and the sum of m over all test cases both do not exceed $2\,000$.

Output

For each testcase, output "YES" if the graph contains a subgraph that is a Fish Graph, otherwise print "NO". If the answer is "YES", on the following lines output a description of the subgraph.

The first line of the description contains one integer k — the number of edges of the subgraph.

On the next k lines, output the edges of the chosen subgraph. Each of the k lines should contains two integers u and v ($1 \leq u, v \leq n$, $u \neq v$) — the edge between u and v belongs to the subgraph. The order in which u and v are printed does not matter, as long as the two nodes are connected by an edge in the original graph. The order in which you print the edges does not matter, as long as the resulting subgraph is a fish graph.

If there are multiple solutions, print any.

input

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

output

```
YES
6
5 4
6 4
4 3
1 4
2 1
3 2
YES
5
5 3
2 3
3 1
4 3
1 4
NO
```

In the first example, a possible valid subgraph contains the cycle $1 \to 2 \to 3 \to 4 \to 1$. The special node of this cycle is node 4. The two extra edges $4 - 5$ and $4 - 6$ are both connected to 4, completing the Fish Graph.

In the second example, a possible valid subgraph contains the cycle $1 \to 3 \to 4 \to 1$. The special node of this cycle is node 3. The two extra edges $3 - 2$ and $3 - 5$ are both connected to 3, completing the Fish Graph.

In the last example, it can be proven that there is no valid subgraph.

C. Similar Polynomials

4 seconds, 256 megabytes

A polynomial $A(x)$ of degree d is an expression of the form $A(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_d x^d$, where a_i are integers, and $a_d \neq 0$. Two polynomials $A(x)$ and $B(x)$ are called similar if there is an integer s such that for any integer x it holds that

$$B(x) \equiv A(x + s) \pmod{10^9 + 7}.$$

For two similar polynomials $A(x)$ and $B(x)$ of degree d, you're given their values in the points $x = 0, 1, \ldots, d$ modulo $10^9 + 7$.

Find a value s such that $B(x) \equiv A(x + s) \pmod{10^9 + 7}$ for all integers x.

Input

The first line contains a single integer d ($1 \le d \le 2\,500\,000$).

The second line contains $d + 1$ integers $A(0), A(1), \ldots, A(d)$ ($0 \le A(i) < 10^9 + 7$) — the values of the polynomial $A(x)$.

The third line contains $d + 1$ integers $B(0), B(1), \ldots, B(d)$ ($0 \le B(i) < 10^9 + 7$) — the values of the polynomial $B(x)$.

It is guaranteed that $A(x)$ and $B(x)$ are similar and that the leading coefficients (i.e., the coefficients in front of x^d) of $A(x)$ and $B(x)$ are not divisible by $10^9 + 7$.

Output

Print a single integer s ($0 \le s < 10^9 + 7$) such that $B(x) \equiv A(x + s) \pmod{10^9 + 7}$ for all integers x.

If there are multiple solutions, print any.

input

```
1
1000000006 0
2 3
```

output

```
3
```

input

```
2
1 4 9
100 121 144
```

output

```
9
```

In the first example, $A(x) \equiv x - 1 \pmod{10^9 + 7}$ and $B(x) \equiv x + 2 \pmod{10^9 + 7}$. They're similar because

$$B(x) \equiv A(x + 3) \pmod{10^9 + 7}.$$

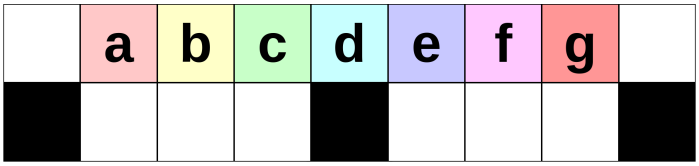
In the second example, $A(x) \equiv (x + 1)^2 \pmod{10^9 + 7}$ and $B(x) \equiv (x + 10)^2 \pmod{10^9 + 7}$, hence

$$B(x) \equiv A(x + 9) \pmod{10^9 + 7}.$$

D. Toy Machine

1 second, 256 megabytes

There is a toy machine with toys arranged in two rows of n cells each (n is odd).



Initial state for $n = 9$.

Initially, $n - 2$ toys are placed in the non-corner cells of the top row. The bottom row is initially empty, and its leftmost, rightmost, and central cells are blocked. There are 4 buttons to control the toy machine: left, right, up, and down marked by the letters `L`, `R`, `U`, and `D` correspondingly.

When pressing `L`, `R`, `U`, or `D`, all the toys will be moved simultaneously in the corresponding direction and will only stop if they push into another toy, the wall or a blocked cell. Your goal is to move the k-th toy into the leftmost cell of the top row. The toys are numbered from 1 to $n - 2$ from left to right. Given n and k, find a solution that uses at most $1\,000\,000$ button presses.

To test out the toy machine, a **web page** is available that lets you play the game in real time.

Input

The first and only line contains two integers, n and k ($5 \le n \le 100\,000$, n is odd, $1 \le k \le n - 2$) — the number of cells in a row, and the index of the toy that has to be moved to the leftmost cell of the top row.

Output

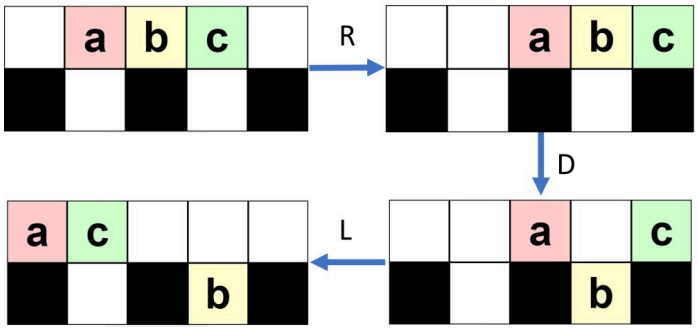
On a single line, output a description of the button presses as a string of at most $1\,000\,000$ characters. The string should only contain the characters `L`, `R`, `U`, and `D`. The i-th character in the string is the i-th button that is pressed. After all the button presses are performed, the k-th toy should be in the leftmost cell of the top row.

If there are multiple solutions, print any. The number of button presses does not have to be minimized.

input
5 1
output
RDL

input
7 2
output
RDL

In the first example, there will be $5 - 2 = 3$ toys. The first toy needs to end up in the leftmost cell of the top row. The moves `RDL` will achieve this, see the picture for a better understanding. Another possible solution would be to do one button press `L`.



Visualization of the moves for the first example.

E. Half-sum

4 seconds, 256 megabytes

You're given a multiset of non-negative integers $\{a_1, a_2, \ldots, a_n\}$.

In one step you take two elements x and y of the multiset, remove them and insert their mean value $\frac{x+y}{2}$ back into the multiset.

You repeat the step described above until you are left with only two numbers A and B. What is the maximum possible value of their absolute difference $|A - B|$?

Since the answer is not an integer number, output it modulo $10^9 + 7$.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 100$). Description of the test cases follows.

The first line of each test case contains a single integer n ($2 \le n \le 10^6$) — the size of the multiset.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$) — the elements of the multiset.

It is guaranteed that the sum of n over all test cases does not exceed 10^6.

Output

For each test case, output a single integer, the answer to the problem modulo $10^9 + 7$.

Formally, let $M = 10^9 + 7$. It can be shown that the answer can be expressed as an irreducible fraction $\frac{p}{q}$, where p and q are integers and $q \not\equiv 0 \pmod{M}$. Output the integer equal to $p \cdot q^{-1} \bmod M$. In other words, output an integer x such that $0 \le x < M$ and $x \cdot q \equiv p \pmod{M}$.

input
5 2 7 3 4 1 2 10 11 3 1 2 3 6 64 32 64 16 64 0 4 1 1 1 1
output
4 9 500000005 59 0

In the first case, you can't do any operations, so the answer is $|7 - 3| = 4$.

In the second case, one of the optimal sequence of operations:

1. Substitute 1 and 2 with 1.5;
2. Substitute 10 and 11 with 10.5;
3. The difference between 1.5 and 10.5 is 9.

In the third case, the exact answer is $\frac{3}{2}$, and $500\,000\,005 \cdot 2 \equiv 3 \pmod{10^9 + 7}$.

F. Entangled Substrings

2 seconds, 256 mebibytes

Quantum entanglement is when two particles link together in a certain way no matter how far apart they are in space.

You are given a string s. A pair of its non-empty substrings (a, b) is called *entangled* if there is a (possibly empty) link string c such that:

- Every occurrence of a in s is immediately followed by cb;
- Every occurrence of b in s is immediately preceded by ac.

In other words, a and b occur in s only as substrings of acb. Compute the total number of entangled pairs of substrings of s.

A string a is a substring of a string b if a can be obtained from b by the deletion of several (possibly zero or all) characters from the beginning and several (possibly zero or all) characters from the end.

Input

The first and only line contains a string s of lowercase English letters ($1 \le |s| \le 10^5$) — the string for which you should count pairs of entangled substrings.

Output

Output a single integer, the number of entangled pairs of substrings of s.

input
abba
output
1

input
abacaba
output
0

input
abcabcabcabc
output
5

input
adamant
output
82

In the first example, the only entangled pair is (ab, ba). For this pair, the corresponding link string c is empty, as they only occur as substrings of the whole string abba, which doesn't have any characters between ab and ba.

In the second example, there are no entangled pairs.

In the third example, the entangled pairs are (a, b), (b, c), (a, c), (a, bc), and (ab, c). For most pairs, the corresponding link string c is empty, except for the pair (a, c), for which the link string c is b, as a and c only occur as substrings of the string abc.