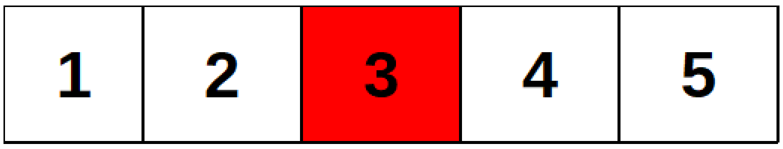
A. Water The Garden

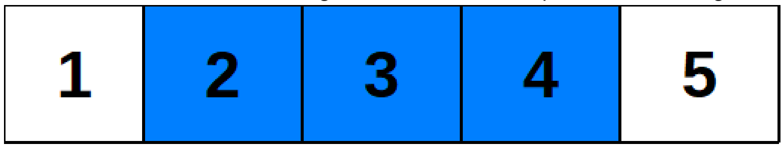
time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

It is winter now, and Max decided it's about time he watered the garden.

The garden can be represented as n consecutive garden beds, numbered from 1 to n. k beds contain water taps (i-th tap is located in the bed x_i), which, if turned on, start delivering water to neighbouring beds. If the tap on the bed x_i is turned on, then after one second has passed, the bed x_i will be watered; after two seconds have passed, the beds from the segment $[x_i - 1, x_i + 1]$ will be watered (if they exist); after j seconds have passed (j **is an integer number**), the beds from the segment $[x_i - (j - 1), x_i + (j - 1)]$ will be watered (if they exist). **Nothing changes during the seconds, so, for example, we can't say that the segment $[x_i - 2.5, x_i + 2.5]$ will be watered after 2.5 seconds have passed; only the segment $[x_i - 2, x_i + 2]$ will be watered at that moment.**



The garden from test 1. White colour denotes a garden bed without a tap, red colour — a garden bed with a tap.



The garden from test 1 after 2 seconds have passed after turning on the tap. White colour denotes an unwatered garden bed, blue colour — a watered bed.

Max wants to **turn on all the water taps at the same moment**, and now he wonders, what is the minimum number of seconds that have to pass after he turns on some taps until the whole garden is watered. Help him to find the answer!

Input

The first line contains one integer t — the number of test cases to solve ($1 \le t \le 200$).

Then t test cases follow. The first line of each test case contains two integers n and k ($1 \le n \le 200$, $1 \le k \le n$) — the number of garden beds and water taps, respectively.

Next line contains k integers x_i ($1 \le x_i \le n$) — the location of i-th water tap. It is guaranteed that for each $i \in [2; k]$ condition $x_{i-1} < x_i$ holds.

It is guaranteed that the sum of n over all test cases doesn't exceed 200.

Note that in hacks you have to set $t = 1$.

Output

For each test case print one integer — the minimum number of seconds that have to pass after Max turns on some of the water taps, until the whole garden is watered.

Example

input

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

output

```
3
1
4
```

Note

The first example consists of 3 tests:

1. There are 5 garden beds, and a water tap in the bed 3. If we turn it on, then after 1 second passes, only bed 3 will be watered; after 2 seconds pass, beds $[1, 3]$ will be watered, and after 3 seconds pass, everything will be watered.

2. There are 3 garden beds, and there is a water tap in each one. If we turn all of them on, then everything will be watered after 1 second passes.
3. There are 4 garden beds, and only one tap in the bed 1. It will take 4 seconds to water, for example, bed 4.

B. Tea Queue

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

Recently n students from city S moved to city P to attend a programming camp.

They moved there by train. In the evening, all students in the train decided that they want to drink some tea. Of course, no two people can use the same teapot simultaneously, so the students had to form a queue to get their tea.

i-th student comes to the end of the queue at the beginning of l_i-th second. If there are multiple students coming to the queue in the same moment, then the student with greater index comes after the student with lesser index. Students in the queue behave as follows: if there is nobody in the queue before the student, then he uses the teapot for exactly one second and leaves the queue with his tea; otherwise the student waits for the people before him to get their tea. If at the beginning of r_i-th second student i still cannot get his tea (there is someone before him in the queue), then he leaves the queue without getting any tea.

For each student determine the second he will use the teapot and get his tea (if he actually gets it).

Input

The first line contains one integer t — the number of test cases to solve ($1 \le t \le 1000$).

Then t test cases follow. The first line of each test case contains one integer n ($1 \le n \le 1000$) — the number of students.

Then n lines follow. Each line contains two integer l_i, r_i ($1 \le l_i \le r_i \le 5000$) — the second i-th student comes to the end of the queue, and the second he leaves the queue if he still cannot get his tea.

It is guaranteed that for every $i \in [2;\ n]$ condition $l_{i-1} \le l_i$ holds.

The sum of n over all test cases doesn't exceed 1000.

Note that in hacks you have to set $t = 1$.

Output

For each test case print n integers. i-th of them must be equal to the second when i-th student gets his tea, or 0 if he leaves without tea.

Example

input

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

output

```
1 2
1 0 2
```

Note

The example contains 2 tests:

1. During 1-st second, students 1 and 2 come to the queue, and student 1 gets his tea. Student 2 gets his tea during 2-nd second.
2. During 1-st second, students 1 and 2 come to the queue, student 1 gets his tea, and student 2 leaves without tea. During 2-nd second, student 3 comes and gets his tea.

C. Swap Adjacent Elements

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

You have an array a consisting of n integers. Each integer from 1 to n appears exactly once in this array.

For some indices i ($1 \le i \le n - 1$) it is possible to swap i-th element with $(i + 1)$-th, for other indices it is not possible. You may perform any number of swapping operations any order. There is no limit on the number of times you swap i-th element with $(i + 1)$-th (if the position is not forbidden).

Can you make this array sorted in ascending order performing some sequence of swapping operations?

Input

The first line contains one integer n $(2 \le n \le 200000)$ — the number of elements in the array.

The second line contains n integers a_1, a_2, ..., a_n $(1 \le a_i \le 200000)$ — the elements of the array. Each integer from 1 to n appears exactly once.

The third line contains a string of n - 1 characters, each character is either `0` or `1`. If i-th character is `1`, then you can swap i-th element with $(i + 1)$-th any number of times, otherwise it is forbidden to swap i-th element with $(i + 1)$-th.

Output

If it is possible to sort the array in ascending order using any sequence of swaps you are allowed to make, print `YES`. Otherwise, print `NO`.

Examples

input
```
6
1 2 5 3 4 6
01110
```
output
```
YES
```

input
```
6
1 2 5 3 4 6
01010
```
output
```
NO
```

Note

In the first example you may swap a_3 and a_4, and then swap a_4 and a_5.

D. Tanks

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Petya sometimes has to water his field. To water the field, Petya needs a tank with exactly V ml of water.

Petya has got N tanks, i-th of them initially containing a_i ml of water. The tanks are really large, any of them can contain any amount of water (no matter how large this amount is).

Also Petya has got a scoop that can contain up to K ml of water (initially the scoop is empty). This scoop can be used to get some water from some tank, and after that pour it all into some tank (it is impossible to get water from multiple tanks without pouring it, or leave some water in the scoop when pouring it). When Petya tries to get some water from a tank, he gets $min(v, K)$ water, where v is the current volume of water in the tank.

Is it possible to obtain a tank with exactly V ml of water using these operations? If it is possible, print a sequence of operations that allows to do it. If there are multiple ways to obtain needed amount of water in some tank, print any of them.

Input

The first line contains 3 integers: N $(2 \le N \le 5000)$, K $(1 \le K \le 5000)$, and V $(0 \le V \le 10^9)$ — the number of tanks, the maximum volume of water the scoop can contain, and the required amount of water in some tank, respectively.

The second line contains N integers a_i $(0 \le a_i \le 10^5)$, where a_i is initial volume of water in i-th tank.

Output

If it is impossible to obtain a tank with exactly V ml of water, print `NO`.

Otherwise print `YES` in the first line, and beginning from the second line, print the sequence of operations in the following format:

Each line has to contain 3 numbers denoting a compressed operation: "$cnt\ x\ y$" $(1 \le cnt \le 10^9,\ 1 \le x,\ y \le N)$, where x is the index of the tank where we get water, y is the index of the tank where we pour water, and cnt is the number of times we transfer water from tank x to tank y.

The number of these lines **must not exceed** $N + 5$.

Examples

input
```
2 3 5
2 3
```
output
```
YES
1 2 1
```

input

```
2 3 4
2 3
```

output

```
NO
```

input

```
5 2 0
1 3 5 7 9
```

output

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

E. Connected Components?

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

You are given an undirected graph consisting of n vertices and $\frac{n(n-1)}{2} - m$ edges. Instead of giving you the edges that exist in the graph, we give you m unordered pairs (x, y) such that there is no edge between x and y, and if some pair of vertices is not listed in the input, then there is an edge between these vertices.

You have to find the number of connected components in the graph and the size of each component. A connected component is a set of vertices X such that for every two vertices from this set there exists at least one path in the graph connecting these vertices, but adding any other vertex to X violates this rule.

Input

The first line contains two integers n and m ($1 \le n \le 200000$, $0 \le m \le min(\frac{n(n-1)}{2}, 200000)$).

Then m lines follow, each containing a pair of integers x and y ($1 \le x, y \le n$, $x \ne y$) denoting that **there is no edge** between x and y. Each pair is listed at most once; (x, y) and (y, x) are considered the same (so they are never listed in the same test). If some pair of vertices is not listed in the input, then there **exists** an edge between those vertices.

Output

Firstly print k — the number of connected components in this graph.

Then print k integers — the sizes of components. You should output these integers in non-descending order.

Example

input

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

output

```
2
1 4
```

F. SUM and REPLACE

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

Let $D(x)$ be the number of positive divisors of a positive integer x. For example, $D(2) = 2$ (2 is divisible by 1 and 2), $D(6) = 4$ (6 is divisible by 1, 2, 3 and 6).

You are given an array a of n integers. You have to process two types of queries:

1. `REPLACE` l r — for every $i \in [l, r]$ replace a_i with $D(a_i)$;
2. `SUM` l r — calculate $\sum\limits_{i=l}^{r} a_i$.

Print the answer for each `SUM` query.

Input

The first line contains two integers n and m ($1 \leq n, m \leq 3 \cdot 10^5$) — the number of elements in the array and the number of queries to process, respectively.

The second line contains n integers $a_1, a_2, ..., a_n$ ($1 \leq a_i \leq 10^6$) — the elements of the array.

Then m lines follow, each containing 3 integers t_i, l_i, r_i denoting i-th query. If $t_i = 1$, then i-th query is `REPLACE` l_i r_i, otherwise it's `SUM` l_i r_i ($1 \leq t_i \leq 2$, $1 \leq l_i \leq r_i \leq n$).

There is at least one `SUM` query.

Output

For each `SUM` query print the answer to it.

Example

input

```
7 6
6 4 1 10 3 2 4
2 1 7
2 4 5
1 3 5
2 4 4
1 5 7
2 1 7
```

output

```
30
13
4
22
```

G. List Of Integers

time limit per test: 5 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Let's denote as $L(x, p)$ an infinite sequence of integers y such that $gcd(p, y) = 1$ and $y > x$ (where gcd is the greatest common divisor of two integer numbers), sorted in ascending order. The elements of $L(x, p)$ are 1-indexed; for example, 9, 13 and 15 are the first, the second and the third elements of $L(7, 22)$, respectively.

You have to process t queries. Each query is denoted by three integers x, p and k, and the answer to this query is k-th element of $L(x, p)$.

Input

The first line contains one integer t ($1 \leq t \leq 30000$) — the number of queries to process.

Then t lines follow. i-th line contains three integers x, p and k for i-th query ($1 \leq x, p, k \leq 10^6$).

Output

Print t integers, where i-th integer is the answer to i-th query.

Examples

input

```
3
7 22 1
7 22 2
7 22 3
```

output

```
9
13
15
```

input

```
5
42 42 42
43 43 43
44 44 44
45 45 45
46 46 46
```

output

```
187
87
```

```
139
128
141
```