



Chicken Farm

After completing his HSC exams, Antar decides that he has had enough of studying and wants to live a peaceful life. He moves to a quiet village in Akibland and starts his very own chicken farm.

However, Antar soon realizes that farming isn't as peaceful as he imagined.

Antar has n food boxes arranged in a line. The i -th box contains $A[i]$ units of food.

For each of his q queries, Antar wants to feed a group of k chickens, where every chicken has a different appetite. He is given a ratio $B[1], B[2], \dots, B[k]$ describing how the food should be divided among these k chickens.

The chickens must receive their food **in order**: the 1st chicken must receive its food before the 2nd, the 2nd before the 3rd, and so on. The food boxes must also be used **in their original order** — Antar cannot rearrange them. Therefore, all n boxes must be split into k **contiguous, non-overlapping segments** (covering every box, in order), and the i -th segment is given entirely to the i -th chicken.

Let $B'[i]$ be the total amount of food actually received by the i -th chicken under such a split. Antar wants this split to satisfy, for every $1 \leq i, j \leq k$:

$$\frac{B'[i]}{B'[j]} = \frac{B[i]}{B[j]}$$

In other words, the food received by the chickens must be split in **exactly** the ratio $B[1] : B[2] : \dots : B[k]$.

For each query, determine whether such a split is possible. If it is, report where each chicken's segment ends.

Input Format

The input is read from standard input in the following format:

- Line 1: two integers n and q — the number of food boxes and the number of queries.
- Line 2: n integers $A[1], A[2], \dots, A[n]$ — the amount of food in each box, in order.
- Each of the next q queries is given on two lines:
 - The first line contains a single integer k — the number of chickens in this query.

- The second line contains k integers $B[1], B[2], \dots, B[k]$ — the desired food ratio for this query.

Output Format

For each query, output:

If it is impossible to split all n boxes into k contiguous segments (covering all boxes, in order) whose totals are in the ratio $B[1] : B[2] : \dots : B[k]$ output the following.

- Line 1: NO

If it is possible, output the following.

- Line 1: YES
- Line 2: The sequence of k numbers separated by spaces.

If it is possible, but you do not output the sequence, output the following:

- Line 1: YES -1

Constraints

- $1 \leq n, q \leq 10^5$
- $1 \leq A[i] \leq 10^9$
- $1 \leq B[i] \leq 10^9$
- $1 \leq k \leq n$
- The sum of k over all queries does not exceed 10^5 .

Subtasks

Subtask	Points	n	q	k	Additional Constraints
0	0	-	-	-	The examples
1	5	≤ 1000	$= 1$	$= 2$	No additional constraints.
2	10	≤ 1000	-	$= 3$	$\sum k \leq 1000$
3	15	$\leq 10^5$	-	$= 2$	No additional constraints.
4	30	$\leq 10^5$	-	$= 3$	No additional constraints.
5	40	$\leq 10^5$	$\leq 10^5$	$\leq n$	No additional constraints.

Scoring

- If you choose not to print the sequence of numbers but provide the correct answer, you will receive 20% of the subtask's score
- If you choose to print the sequence, but it is incorrect, you will receive 0 points for that subtask.
- Otherwise, you will receive full score for the subtask.

Example

```
6 5
1 12 13 13 51 1
3
1 2 4
3
2 1 4
3
1 1 5
3
4 2 1
3
1 38 52
```

The partially correct output is:

```
YES -1
YES -1
YES -1
NO
YES -1
```

The correct output is:

YES
2 4 6
YES
3 4 6
YES
2 3 6
NO
YES
1 4 6

Explanation

Consider the first query:

$$1 : 2 : 4$$

Antar can divide the boxes as follows:

- The 1st chicken receives boxes 1..2 $\rightarrow 1 + 12 = 13$
- The 2nd chicken receives boxes 3..4 $\rightarrow 13 + 13 = 26$
- The 3rd chicken receives boxes 5..6 $\rightarrow 51 + 1 = 52$

Therefore,

$$13 : 26 : 52 = 1 : 2 : 4.$$

So the distribution is possible, and the last box received by each chicken is:

2 4 6

Hence the answer is:

YES
2 4 6

For the fourth query (4 : 2 : 1), no way of splitting the 6 boxes into 3 contiguous, ordered segments produces totals in that exact ratio, so the answer is NO.



Dictator Pepe

The days of jokes are over!

Our beloved leader **Pepe** has become the supreme leader of Pepeland. As his first great decision, he has decided to eliminate all the noobs and fools from Pepeland. Only the smartest ones may remain!

To find out who is truly smart enough to stay, Pepe decides to create the riddle of the century. For this, he seeks the help of the smartest person on Earth, **Woumya**.

Pepe takes Woumya to a huge dungeon in Pepeland, where **Dali** and **Sesh** are trapped. The dungeon consists of n rooms connected by corridors. Each room has a number written on it, where the number written on room i is A_i .

A **path** in the dungeon from room u to v is defined as a sequence of distinct rooms $p_1, p_2, \dots, p_m$ such that $p_1 = u$, $p_m = v$ and p_i, p_{i+1} is directly connected by a corridor for all $1 \leq i < m$.

Woumya can choose any path through the dungeon. Suppose the path contains the rooms

$$u_1, u_2, \dots, u_k$$

in this order.

The **beauty** of this path is defined as

$$\sum_{i=1}^k \gcd(A_{u_1}, A_{u_2}, \dots, A_{u_i}).$$

Woumya wants to impress Pepe by finding the path with the **maximum possible beauty**. A path containing only a single room is also allowed.

Your task is to calculate the maximum possible beauty.

Input Format

The input is read from standard input in the following format:

Each test file contains multiple test cases. **The first line of the input contains the number of test cases T** . Then, for each test case, the input is given in the following format:

- Line 1: An integer n — the number of rooms in the dungeon.
- Line 2: n integers $A_1, A_2, \dots, A_n$ — the numbers written on the rooms.
- Each of the next $n - 1$ lines contains two integers u and v , meaning that rooms u and v are directly connected by a corridor.

Output Format

For each test case, output a single integer — the **maximum possible beauty** of any route through the dungeon.

Constraints

- $1 \leq T \leq 10^4$
- $1 \leq n \leq 10^5$
- $1 \leq A_i \leq 10^4$ for all $1 \leq i \leq n$
- $1 \leq u, v \leq n$
- It is guaranteed that there exists a unique path between any pair of rooms.
- The sum of n over all test cases does not exceed 10^5 .

Subtasks

Subtask	Points	n	A_i	Additional Constraints
0	0	-	-	The examples
1	10	$\leq 10^5$	$\leq 10^4$	Every room is directly connected to a room r by a corridor where $1 \leq r \leq n$.
2	15	$\leq 10^4$	$\leq 10^4$	No additional constraints.
3	75	$\leq 10^5$	$\leq 10^4$	No additional constraints.

Example

Example 1

Input

```
1
4
29 26 19 28
1 2
2 3
3 4
```

Output

```
32
```

Explanation

The optimal path is from room 1 to 4 where the rooms are 1, 2, 3, 4 in order. Hence, the maximum **beauty** is $\gcd(A_1) + \gcd(A_1, A_2) + \gcd(A_1, A_2, A_3) + \gcd(A_1, A_2, A_3, A_4) = \gcd(29) + \gcd(29, 26) + \gcd(29, 26, 19) + \gcd(29, 26, 19, 28) = 32$

Example 2

Input

```
1
6
7 10 2 15 9 6
1 2
1 3
1 5
3 4
3 6
```

Output

```
18
```

Explanation

The optimal path is from room 4 to 2 where the rooms are 4, 3, 1, 2 in order. Hence, the maximum beauty is $\gcd(A_4) + \gcd(A_4, A_3) + \gcd(A_4, A_3, A_1) + \gcd(A_4, A_3, A_1, A_2) = \gcd(15) + \gcd(15, 2) + \gcd(15, 2, 7) + \gcd(15, 2, 7, 10) = 18$

Example 3

Input

```
3
6
4669 9654 8331 3885 9651 7806
1 2
2 3
3 4
4 5
1 6
10
89 23 34 72 41 3 60 99 48 97
1 2
1 3
3 4
4 5
2 6
6 7
3 8
2 9
7 10
4
79 26 19 78
1 2
2 3
3 4
```

Output

```
9663
105
82
```



Orders

Turjo is managing the front counter at Akibdonalds. He has exactly N pagers on his charging rack, numbered from 1 to N .

A group of k customers ($k \geq 1$) must receive k **consecutive pager IDs** so that Turjo can carry them on one tray.

For every arriving group, Turjo chooses the **smallest possible starting ID** l such that all pagers

$$l, l + 1, \dots, l + k - 1$$

are currently available.

When the customers' food is ready, they return their pagers to Turjo, making those pagers available again. The returned pagers always form a consecutive range.

You are given a sequence of Q events in chronological order. Simulate Turjo's actions and determine which pager IDs each arriving group receives.

Input Format

The first line contains two integers:

- line 1: $N \ Q$

The next Q lines describe the following events.

- line $1 + i$ ($1 \leq i \leq Q$):
 - 1 k — A group of k customers arrives and places an order. It is guaranteed that $1 \leq k \leq N$.
 - 2 $l \ r$ — Customers return pagers l through r , making all of them available again. It is guaranteed that $1 \leq l \leq r \leq N$, and every pager in this range is currently being held by a customer.

All group arrivals can be fulfilled using the currently available pagers.

Output Format

For every type 1 event:

- If $k = 1$, print one integer: the **ID** of the pager assigned to the customer.
- If $k > 1$, print two integers l and r : the first and last pager IDs assigned to the group.

Print the answers in chronological order.

Constraints

- $1 \leq N \leq 200,000$
- $1 \leq Q \leq 200,000$
- $1 \leq k \leq N$
- $1 \leq l \leq r \leq N$

Subtasks

Subtask	Points	N	Q	k	Additional Constraints
0	0	-	-	-	The examples
1	5	$= 1$	$\leq 200,000$	$= 1$	All events involve the only pager.
2	15	$\leq 5,000$	$\leq 5,000$	$1 \leq k \leq N$	No additional constraints.
3	15	$\leq 200,000$	$\leq 200,000$	$= 1$	There are no group orders. Formally, there are no Type 2 queries.
4	25	$\leq 200,000$	$\leq 200,000$	≤ 3	No additional constraints.
5	40	$\leq 200,000$	$\leq 200,000$	$1 \leq k \leq N$	No additional constraints.

Examples

Example 1

Input:

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

Output:

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

Explanation

There are 6 pagers, numbered from 1 to 6.

We process the events one by one:

1. In the first event, one customer arrives and receives pager 1.
2. In the second event, a group of 2 customers arrives. Pagers 2 and 3 are the smallest consecutive available pagers, so they receive:

2, 3

3. In the third event, the customer using pager 1 returns it. Pager 1 becomes available again.
4. In the fourth event, a group of 2 customers arrives. Pagers 4 and 5 are the smallest consecutive available pagers, so they receive:

4, 5

5. In the fifth event, pagers 4 through 5 are returned. Therefore, both pagers 4 and 5 become available again.
6. In the sixth event, a group of 3 customers arrives.

At this moment, the available pagers are:

1, 4, 5, 6

The smallest three consecutive available pagers are 4, 5, 6, so the group receives:

4, 5, 6

Therefore, we print:

```
4 6
```

7. In the final event, one customer arrives. Pager 1 is the smallest available pager, so the customer receives pager 1.

Thus the complete output is:

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