Codeforces Round #381 (Div. 2)

A. Alyona and copybooks

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

Little girl Alyona is in a shop to buy some copybooks for school. She study four subjects so she wants to have equal number of copybooks for each of the subjects. There are three types of copybook's packs in the shop: it is possible to buy one copybook for a rubles, a pack of two copybooks for b rubles, and a pack of three copybooks for c rubles. Alyona already has n copybooks.

What is the minimum amount of rubles she should pay to buy such number of copybooks k that $n + k$ is divisible by 4? There are infinitely many packs of any type in the shop. Alyona can buy packs of different type in the same purchase.

Input

The only line contains 4 integers n, a, b, c ($1 \leq n, a, b, c \leq 10^9$).

Output

Print the minimum amount of rubles she should pay to buy such number of copybooks k that $n + k$ is divisible by 4.

Examples

input
```
1 1 3 4
```
output
```
3
```

input
```
6 2 1 1
```
output
```
1
```

input
```
4 4 4 4
```
output
```
0
```

input
```
999999999 1000000000 1000000000 1000000000
```
output
```
1000000000
```

Note

In the first example Alyona can buy 3 packs of 1 copybook for $3a = 3$ rubles in total. After that she will have 4 copybooks which she can split between the subjects equally.

In the second example Alyuna can buy a pack of 2 copybooks for $b = 1$ ruble. She will have 8 copybooks in total.

In the third example Alyona can split the copybooks she already has between the 4 subject equally, so she doesn't need to buy anything.

In the fourth example Alyona should buy one pack of one copybook.

B. Alyona and flowers

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

Little Alyona is celebrating Happy Birthday! Her mother has an array of n flowers. Each flower has some mood, the mood of i-th flower is a_i. The mood can be positive, zero or negative.

Let's define a subarray as a segment of consecutive flowers. The mother suggested some set of subarrays. Alyona wants to choose several of the subarrays suggested by her mother. After that, each of the flowers will add to the girl's happiness its mood multiplied by the number of chosen subarrays the flower is in.

For example, consider the case when the mother has 5 flowers, and their moods are equal to $1, -2, 1, 3, -4$. Suppose the mother suggested subarrays $(1, -2)$, $(3, -4)$, $(1, 3)$, $(1, -2, 1, 3)$. Then if the girl chooses the third and the fourth subarrays then:

- the first flower adds $1 \cdot 1 = 1$ to the girl's happiness, because he is in one of chosen subarrays,
- the second flower adds $(-2) \cdot 1 = -2$, because he is in one of chosen subarrays,
- the third flower adds $1 \cdot 2 = 2$, because he is in two of chosen subarrays,
- the fourth flower adds $3 \cdot 2 = 6$, because he is in two of chosen subarrays,
- the fifth flower adds $(-4) \cdot 0 = 0$, because he is in no chosen subarrays.

Thus, in total $1 + (-2) + 2 + 6 + 0 = 7$ is added to the girl's happiness. Alyona wants to choose such subarrays from those suggested by the mother that the value added to her happiness would be as large as possible. Help her do this!

Alyona can choose any number of the subarrays, even 0 or all suggested by her mother.

Input

The first line contains two integers n and m $(1 \le n, m \le 100)$ — the number of flowers and the number of subarrays suggested by the mother.

The second line contains the flowers moods — n integers $a_1, a_2, ..., a_n$ $(-100 \le a_i \le 100)$.

The next m lines contain the description of the subarrays suggested by the mother. The i-th of these lines contain two integers l_i and r_i $(1 \le l_i \le r_i \le n)$ denoting the subarray $a[l_i], a[l_i + 1], ..., a[r_i]$.

Each subarray can encounter more than once.

Output

Print single integer — the maximum possible value added to the Alyona's happiness.

Examples

input

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

output

```
7
```

input

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

output

```
16
```

input

```
2 2
-1 -2
1 1
1 2
```

output

```
0
```

Note

The first example is the situation described in the statements.

In the second example Alyona should choose all subarrays.

The third example has answer 0 because Alyona can choose none of the subarrays.

C. Alyona and mex

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

Alyona's mother wants to present an array of n non-negative integers to Alyona. The array should be special.

Alyona is a capricious girl so after she gets the array, she inspects m of its subarrays. Subarray is a set of some subsequent elements of the array. The i-th subarray is described with two integers l_i and r_i, and its elements are $a[l_i], a[l_i + 1], ..., a[r_i]$.

Alyona is going to find *mex* for each of the chosen subarrays. Among these m *mexes* the girl is going to find the smallest. She wants this minimum *mex* to be as large as possible.

You are to find an array a of n elements so that the minimum *mex* among those chosen by Alyona subarrays is as large as possible.

The *mex* of a set S is a minimum possible non-negative integer that is not in S.

Input

The first line contains two integers n and m $(1 \le n, m \le 10^5)$.

The next m lines contain information about the subarrays chosen by Alyona. The i-th of these lines contains two integers l_i and r_i $(1 \le l_i \le r_i \le n)$, that describe the subarray $a[l_i], a[l_i + 1], ..., a[r_i]$.

Output

In the first line print single integer — the maximum possible minimum *mex*.

In the second line print n integers — the array a. All the elements in a should be between 0 and 10^9.

It is guaranteed that there is an optimal answer in which all the elements in a are between 0 and 10^9.

If there are multiple solutions, print any of them.

Examples

input

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

output

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

input

```
4 2
1 4
2 4
```

output

```
3
5 2 0 1
```

Note

The first example: the *mex* of the subarray $(1, 3)$ is equal to 3, the *mex* of the subarray $(2, 5)$ is equal to 3, the *mex* of the subarray $(4, 5)$ is equal to 2 as well, thus the minumal *mex* among the subarrays chosen by Alyona is equal to 2.

D. Alyona and a tree

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

Alyona has a tree with n vertices. The root of the tree is the vertex 1. In each vertex Alyona wrote an positive integer, in the vertex i she wrote a_i. Moreover, the girl wrote a positive integer to every edge of the tree (possibly, different integers on different edges).

Let's define $dist(v, u)$ as the sum of the integers written on the edges of the simple path from v to u.

The vertex v controls the vertex u ($v \neq u$) if and only if u is in the subtree of v and $dist(v, u) \leq a_u$.

Alyona wants to settle in some vertex. In order to do this, she wants to know for each vertex v what is the number of vertices u such that v controls u.

Input

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

The second line contains n integers $a_1, a_2, ..., a_n$ ($1 \leq a_i \leq 10^9$) — the integers written in the 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 number written on the edge between p_i and $(i + 1)$.

It is guaranteed that the given graph is a tree.

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

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

output

```
1 0 1 0 0
```

input

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

output

```
4 3 2 1 0
```

Note

In the example test case the vertex 1 controls the vertex 3, the vertex 3 controls the vertex 5 (note that is doesn't mean the vertex 1 controls the vertex 5).

E. Alyona and towers

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

Alyona has built n towers by putting small cubes some on the top of others. Each cube has size $1 \times 1 \times 1$. A tower is a non-zero amount of cubes standing on the top of each other. The towers are next to each other, forming a row.

Sometimes Alyona chooses some segment towers, and put on the top of each tower several cubes. Formally, Alyouna chooses some segment of towers from l_i to r_i and adds d_i cubes on the top of them.

Let the sequence $a_1, a_2, ..., a_n$ be the heights of the towers from left to right. Let's call as a segment of towers $a_l, a_{l+1}, ..., a_r$ a hill if the following condition holds: there is integer k ($l \le k \le r$) such that $a_l < a_{l+1} < a_{l+2} < ... < a_k > a_{k+1} > a_{k+2} > ... > a_r$.

After each addition of d_i cubes on the top of the towers from l_i to r_i, Alyona wants to know the maximum width among all hills. The width of a hill is the number of towers in it.

Input

The first line contain single integer n ($1 \le n \le 3 \cdot 10^5$) — the number of towers.

The second line contain n integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^9$) — the number of cubes in each tower.

The third line contain single integer m ($1 \le m \le 3 \cdot 10^5$) — the number of additions.

The next m lines contain 3 integers each. The i-th of these lines contains integers l_i, r_i and d_i ($1 \le l \le r \le n$, $1 \le d_i \le 10^9$), that mean that Alyona puts d_i cubes on the tio of each of the towers from l_i to r_i.

Output

Print m lines. In i-th line print the maximum width of the hills after the i-th addition.

Example

input

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

output

```
2
4
5
```

Note

The first sample is as follows:

After addition of 2 cubes on the top of each towers from the first to the third, the number of cubes in the towers become equal to $[7, 7, 7, 5, 5]$. The hill with maximum width is $[7, 5]$, thus the maximum width is 2.

After addition of 1 cube on the second tower, the number of cubes in the towers become equal to $[7, 8, 7, 5, 5]$. The hill with maximum width is now $[7, 8, 7, 5]$, thus the maximum width is 4.

After addition of 1 cube on the fourth tower, the number of cubes in the towers become equal to $[7, 8, 7, 6, 5]$. The hill with maximum width is now $[7, 8, 7, 6, 5]$, thus the maximum width is 5.