Codeforces Round 969 (Div. 2)

A. Dora's Set

1 second, 256 megabytes

Dora has a set s containing integers. In the beginning, she will put all integers in $[l, r]$ into the set s. That is, an integer x is initially contained in the set if and only if $l \leq x \leq r$. Then she allows you to perform the following operations:

- Select three **distinct** integers a, b, and c from the set s, such that $\gcd(a, b) = \gcd(b, c) = \gcd(a, c) = 1$[†].
- Then, remove these three integers from the set s.

What is the maximum number of operations you can perform?

[†] Recall that $\gcd(x, y)$ means the greatest common divisor of integers x and y.

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \leq t \leq 500$) — the number of test cases. The description of the test cases follows.

The only line of each test case contains two integers l and r ($1 \leq l \leq r \leq 1000$) — the range of integers in the initial set.

Output

For each test case, output a single integer — the maximum number of operations you can perform.

input
8 1 3 3 7 10 21 2 8 51 60 2 15 10 26 1 1000

output
1 1 3 1 2 3 4 250

In the first test case, you can choose $a = 1$, $b = 2$, $c = 3$ in the only operation, since $\gcd(1, 2) = \gcd(2, 3) = \gcd(1, 3) = 1$, and then there are no more integers in the set, so no more operations can be performed.

In the second test case, you can choose $a = 3$, $b = 5$, $c = 7$ in the only operation.

In the third test case, you can choose $a = 11$, $b = 19$, $c = 20$ in the first operation, $a = 13$, $b = 14$, $c = 15$ in the second operation, and $a = 10$, $b = 17$, $c = 21$ in the third operation. After the three operations, the set s contains the following integers: 12, 16, 18. It can be proven that it's impossible to perform more than 3 operations.

B. Index and Maximum Value

1 second, 256 megabytes

After receiving yet another integer array $a_1, a_2, \ldots, a_n$ at her birthday party, Index decides to perform some operations on it.

Formally, there are m operations that she is going to perform in order. Each of them belongs to one of the two types:

- `+ l r`. Given two integers l and r, for all $1 \leq i \leq n$ such that $l \leq a_i \leq r$, set $a_i := a_i + 1$.
- `- l r`. Given two integers l and r, for all $1 \leq i \leq n$ such that $l \leq a_i \leq r$, set $a_i := a_i - 1$.

For example, if the initial array $a = [7, 1, 3, 4, 3]$, after performing the operation `+ 2 4`, the array $a = [7, 1, 4, 5, 4]$. Then, after performing the operation `- 1 10`, the array $a = [6, 0, 3, 4, 3]$.

Index is curious about the maximum value in the array a. Please help her find it after each of the m operations.

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \leq t \leq 2 \cdot 10^4$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains two integers n and m ($1 \leq n \leq 10^5$, $1 \leq m \leq 10^5$) — the length of the array and the number of operations.

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

Then m lines follow, each line corresponds to the operation, in the following format: `c l r` ($c \in \{+, -\}$, l and r are integers, $1 \leq l \leq r \leq 10^9$) — the description of the operation.

Note that the elements a_i **may not satisfy** $1 \leq a_i \leq 10^9$ after some operations, as it is shown in the example.

It is guaranteed that the sum of n over all test cases does not exceed 10^5, and the sum of m over all test cases does not exceed 10^5.

Output

For each test case, output one single line containing m integers, with the i-th of them describing the maximum value of the array after the i-th operation.

input
5 5 5 1 2 3 2 1 + 1 3 - 2 3 + 1 2 + 2 4 - 6 8 5 5 1 3 3 4 5 + 1 4 + 2 3 - 4 5 - 3 3 - 2 6 5 5 1 1 1 1 1 + 2 3 - 4 5 + 1 6 - 2 5 + 1 8 1 1 1 - 1 1 1 1 1000000000 + 1000000000 1000000000

output
4 4 4 5 5 5 5 4 4 3 1 1 2 1 2 0 1000000001

In the first test case, the process of the operations is listed below:

- After the first operation, the array becomes equal $[2, 3, 4, 3, 2]$. The maximum value is 4.
- After the second operation, the array becomes equal $[1, 2, 4, 2, 1]$. The maximum value is 4.
- After the third operation, the array becomes equal $[2, 3, 4, 3, 2]$. The maximum value is 4.
- After the fourth operation, the array becomes equal $[3, 4, 5, 4, 3]$. The maximum value is 5.
- After the fifth operation, the array becomes equal $[3, 4, 5, 4, 3]$. The maximum value is 5.

In the second test case, the process of the operations is listed below:

- After the first operation, the array becomes equal $[2, 4, 4, 5, 5]$. The maximum value is 5.
- After the second operation, the array becomes equal $[3, 4, 4, 5, 5]$. The maximum value is 5.
- After the third operation, the array becomes equal $[3, 3, 3, 4, 4]$. The maximum value is 4.
- After the fourth operation, the array becomes equal $[2, 2, 2, 4, 4]$. The maximum value is 4.
- After the fifth operation, the array becomes equal $[1, 1, 1, 3, 3]$. The maximum value is 3.

C. Dora and C++

2 seconds, 256 megabytes

Dora has just learned the programming language C++!

However, she has completely misunderstood the meaning of C++. She considers it as two kinds of adding operations on the array c with n elements. Dora has two integers a and b. In one operation, she can choose one of the following things to do.

- Choose an integer i such that $1 \le i \le n$, and increase c_i by a.
- Choose an integer i such that $1 \le i \le n$, and increase c_i by b.

Note that a and b are **constants**, and they can be the same.

Let's define a *range* of array d as $\max(d_i) - \min(d_i)$. For instance, the range of the array $[1, 2, 3, 4]$ is $4 - 1 = 3$, the range of the array $[5, 2, 8, 2, 2, 1]$ is $8 - 1 = 7$, and the range of the array $[3, 3, 3]$ is $3 - 3 = 0$.

After any number of operations (possibly, 0), Dora calculates the range of the new array. You need to help Dora minimize this value, but since Dora loves exploring all by herself, you only need to tell her the minimized value.

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases. The description of test cases follows.

The first line of each test case contains three integers n, a, and b ($1 \le n \le 10^5$, $1 \le a, b \le 10^9$) — the length of the array c and the constant values, respectively.

The second line of each test case contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le 10^9$) — the initial elements of the array c.

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, output a single integer — the minimum possible range of the array after any number of operations.

input

```
10
4 5 5
1 3 4 4
4 2 3
1 3 4 6
4 7 7
1 1 2 6
3 15 9
1 9 5
3 18 12
1 4 5
7 27 36
33 13 23 12 35 24 41
10 6 9
15 5 6 9 8 2 12 15 3 8
2 1 1000000000
1 1000000000
6 336718728 709848696
552806726 474775724 15129785 371139304 178408298 13106071
6 335734893 671469786
138885253 70095920 456876775 9345665 214704906 375508929
```

output

```
3
0
3
2
3
5
1
0
17
205359241
```

In the first test case, we can increase $c_1 = 1$ by $a = 5$. The array c will become $[6, 3, 4, 4]$, and the range is 3. Note that there is more than one way to reach the answer.

In the second test case, we can increase $c_1 = 1$ by $a = 2$ and then increase $c_1 = 3$ by $b = 3$. Also, we can increase $c_2 = 3$ by $b = 3$ and increase $c_3 = 4$ by $a = 2$. The array c will become $[6, 6, 6, 6]$, and the range is 0.

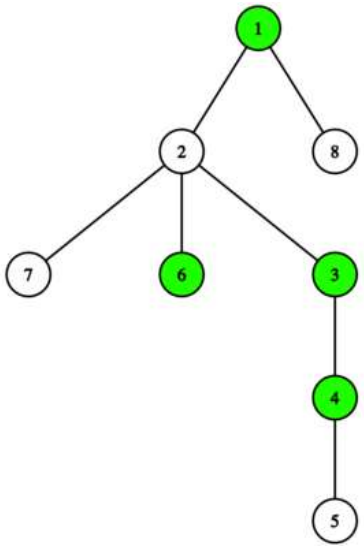
D. Iris and Game on the Tree

2 seconds, 256 megabytes

Iris has a tree rooted at vertex 1. Each vertex has a value of 0 or 1.

Let's consider a leaf of the tree (the vertex 1 is never considered a leaf) and define its *weight*. Construct a string formed by the values of the vertices on the path starting at the root and ending in this leaf. Then the weight of the leaf is the difference between the number of occurrences of 10 and 01 substrings in it.

Take the following tree as an example. Green vertices have a value of 1 while white vertices have a value of 0.



- Let's calculate the weight of the leaf 5: the formed string is 10110. The number of occurrences of substring 10 is 2, the number of occurrences of substring 01 is 1, so the difference is $2 - 1 = 1$.
- Let's calculate the weight of the leaf 6: the formed string is 101. The number of occurrences of substring 10 is 1, the number of occurrences of substring 01 is 1, so the difference is $1 - 1 = 0$.

The *score* of a tree is defined as the number of leaves with non-zero weight in the tree.

But the values of some vertices haven't been decided and will be given to you as $?$. Filling the blanks would be so boring, so Iris is going to invite Dora to play a game. On each turn, one of the girls chooses any of the remaining vertices with value $?$ and changes its value to 0 or 1, **with Iris going first**. The game continues until there are no vertices with value $?$ left in the tree. Iris aims to maximize the score of the tree, while Dora aims to minimize that.

Assuming that both girls play optimally, please determine the final score of the tree.

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 5 \cdot 10^4$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains a single integer n ($2 \le n \le 10^5$) — the number of vertices in the tree.

The following $n-1$ lines each contain two integers u and v ($1 \le u, v \le n$) — denoting an edge between vertices u and v.

It's guaranteed that the given edges form a tree.

The last line contains a string s of length n. The i-th character of s represents the value of vertex i. It's guaranteed that s only contains characters `0`, `1` and `?`.

It is guaranteed that the sum of n over all test cases doesn't exceed $2 \cdot 10^5$.

Output

For each test case, output a single integer — the final score of the tree.

input

```
6
4
1 2
1 3
4 1
0101
4
1 2
3 2
2 4
???0
5
1 2
1 3
2 4
2 5
?1?01
6
1 2
2 3
3 4
5 3
3 6
?0????
5
1 2
1 3
1 4
1 5
11?1?
2
2 1
??
```

output

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

In the first test case, all the values of the vertices have been determined. There are three different paths from the root to a leaf:

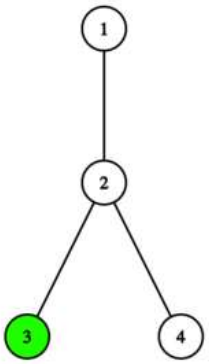
- From vertex 1 to vertex 2. The string formed by the path is `01`, so the weight of the leaf is $0-1=-1$.
- From vertex 1 to vertex 3. The string formed by the path is `00`, so the weight of the leaf is $0-0=0$.
- From vertex 1 to vertex 4. The string formed by the path is `01`, so the weight of the leaf is $0-1=-1$.

Thus, there are two leaves with non-zero weight, so the score of the tree is 2.

In the second test case, one of the sequences of optimal choices for the two players can be:

- Iris chooses to change the value of the vertex 3 to `1`.
- Dora chooses to change the value of the vertex 1 to `0`.
- Iris chooses to change the value of the vertex 2 to `0`.

The final tree is as follows:



The only leaf with non-zero weight is 3, so the score of the tree is 1. Note that this may not be the only sequence of optimal choices for Iris and Dora.

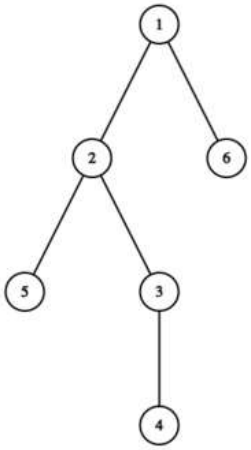
E. Iris and the Tree

3 seconds, 256 megabytes

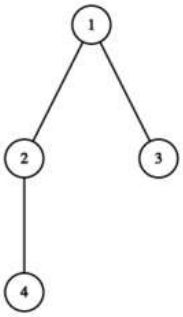
Given a rooted tree with the root at vertex 1. For any vertex i ($1 < i \le n$) in the tree, there is an edge connecting vertices i and p_i ($1 \le p_i < i$), with a weight equal to t_i.

Iris does not know the values of t_i, but she knows that $\sum_{i=2}^{n} t_i = w$ and each of the t_i is a **non-negative integer**.

The vertices of the tree are numbered in a special way: the numbers of the vertices in each subtree are consecutive integers. In other words, the vertices of the tree are numbered in the order of a depth-first search.



The tree in this picture satisfies the condition. For example, in the subtree of vertex 2, the vertex numbers are $2, 3, 4, 5$, which are consecutive integers.



The tree in this picture does not satisfy the condition, as in the subtree of vertex 2, the vertex numbers 2 and 4 are not consecutive integers.

We define $\operatorname{dist}(u, v)$ as the length of the simple path between vertices u and v in the tree.

Next, there will be $n-1$ events:

- Iris is given integers x and y, indicating that $t_x = y$.

After each event, Iris wants to know the maximum possible value of $\operatorname{dist}(i, i \bmod n + 1)$ **independently** for each i ($1 \le i \le n$). She only needs to know the sum of these n values. Please help Iris quickly get the answers.

Note that when calculating the maximum possible values of $\operatorname{dist}(i, i \bmod n + 1)$ and $\operatorname{dist}(j, j \bmod n + 1)$ for $i \neq j$, the unknown edge weights **may be different**.

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains two integers n and w ($2 \leq n \leq 2 \cdot 10^5$, $0 \leq w \leq 10^{12}$) — the number of vertices in the tree and the sum of the edge weights.

The second line of each test case contains $n - 1$ integers $p_2, p_3, \ldots, p_n$ ($1 \leq p_i < i$) — the description of the edges of the tree.

Then follow $n - 1$ lines indicating the events. Each line contains two integers x and y ($2 \leq x \leq n$, $0 \leq y \leq w$), indicating that $t_x = y$.

It is guaranteed that all x in the events are distinct. It is also guaranteed that the sum of all y equals w.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output one line containing $n - 1$ integers, each representing the answer after each event.

input

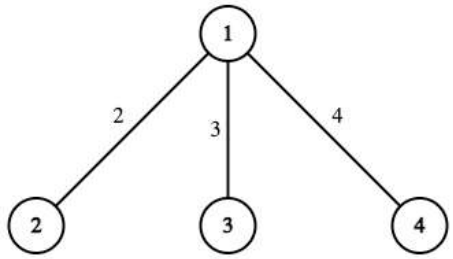
```
4
2 1000000000000
1
2 1000000000000
4 9
1 1 1
2 2
4 4
3 3
6 100
1 2 3 2 1
6 17
3 32
2 4
4 26
5 21
10 511
1 2 2 4 2 1 1 8 8
3 2
6 16
10 256
9 128
2 1
5 8
8 64
4 4
7 32
```

output

```
2000000000000
25 18 18
449 302 247 200 200
4585 4473 2681 1567 1454 1322 1094 1022 1022
```

In the first test case, $\operatorname{dist}(1, 2) = \operatorname{dist}(2, 1) = t_2 = w = 10^{12}$, so $\operatorname{dist}(1, 2) + \operatorname{dist}(2, 1) = 2 \cdot 10^{12}$.

In the second test case, the tree after Iris found out all t_x is shown below:



$\operatorname{dist}(1, 2) = t_2$, $\operatorname{dist}(2, 3) = t_2 + t_3$, $\operatorname{dist}(3, 4) = t_3 + t_4$, $\operatorname{dist}(4, 1) = t_4$. After the first event, she found out that $t_2 = 2$, so $\operatorname{dist}(1, 2) = 2$. At the same time:

- $\operatorname{dist}(2, 3)$ is maximized if $t_3 = 7$, $t_4 = 0$. Then $\operatorname{dist}(2, 3) = 9$.
- $\operatorname{dist}(3, 4)$ and $\operatorname{dist}(4, 1)$ are maximized if $t_3 = 0$, $t_4 = 7$. Then $\operatorname{dist}(3, 4) = \operatorname{dist}(4, 1) = 7$.

Thus, the answer is $2 + 9 + 7 + 7 = 25$.

After the second event, she found out that $t_4 = 4$, then $t_3 = w - t_2 - t_4 = 4$. $\operatorname{dist}(1, 2) = 2$, $\operatorname{dist}(2, 3) = 2 + 3 = 5$, $\operatorname{dist}(3, 4) = 3 + 4 = 7$, $\operatorname{dist}(4, 1) = 4$. Thus, the answer is $2 + 5 + 7 + 4 = 18$.

F. Eri and Expanded Sets

3 seconds, 512 megabytes

Let there be a set that contains **distinct** positive integers. To expand the set to contain as many integers as possible, Eri can choose two integers $x \neq y$ from the set such that their average $\frac{x+y}{2}$ is still a positive integer and isn't contained in the set, and add it to the set. The integers x and y remain in the set.

Let's call the set of integers *consecutive* if, after the elements are sorted, the difference between any pair of adjacent elements is 1. For example, sets $\{2\}$, $\{2, 5, 4, 3\}$, $\{5, 6, 8, 7\}$ are consecutive, while $\{2, 4, 5, 6\}$, $\{9, 7\}$ are not.

Eri likes consecutive sets. Suppose there is an array b, then Eri puts all elements in b into the set. If after a finite number of operations described above, the set can become consecutive, the array b will be called *brilliant*.

Note that if the same integer appears in the array multiple times, we only put it into the set **once**, as a set always contains distinct positive integers.

Eri has an array a of n positive integers. Please help him to count the number of pairs of integers (l, r) such that $1 \leq l \leq r \leq n$ and the subarray $a_l, a_{l+1}, \ldots, a_r$ is brilliant.

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \leq n \leq 4 \cdot 10^5$) — length of the array a.

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

It is guaranteed that the sum of n over all test cases doesn't exceed $4 \cdot 10^5$.

Output

For each test case, output a single integer — the number of brilliant subarrays.

input

```
6
2
2 2
6
1 3 6 10 15 21
5
6 30 18 36 9
1
1000000000
6
1 1 4 5 1 4
12
70 130 90 90 90 108 612 500 451 171 193 193
```

output

```
3
18
5
1
18
53
```

In the first test case, the array $a = [2, 2]$ has 3 subarrays: $[2]$, $[2]$, and $[2, 2]$. For all of them, the set only contains a single integer 2, therefore it's always consecutive. All these subarrays are brilliant, so the answer is 3.

In the second test case, let's consider the subarray $[3, 6, 10]$. We can do operations as follows:

$$\{3, 6, 10\} \xrightarrow{x=6,y=10} \{3, 6, 8, 10\} \xrightarrow{x=6,y=8} \{3, 6, 7, 8, 10\} \xrightarrow{x=3,y=7} \{3, 5, 6, 7, \xrightarrow{x=3,y=5} \{3, 4, 5, 6, 7, 8, 10\} \xrightarrow{x=8,y=10} \{3, 4, 5, 6, 7, 8, 9, 10\}$$

$\{3, 4, 5, 6, 7, 8, 9, 10\}$ is a consecutive set, so the subarray $[3, 6, 10]$ is brilliant.
