Building Bridges

Time Limit: 3 s *Memory Limit: 128 MB*

A wide river has n pillars of possibly different heights standing out of the water. They are arranged in a straight line from one bank to the other. We would like to build a bridge that uses the pillars as support. To achieve this we will select a subset of pillars and connect their tops into sections of a bridge. The subset has to include the first and the last pillar.

The cost of building a bridge section between pillars i and j is $(h_i - h_j)^2$ as we want to avoid uneven sections, where h_i is the height of the pillar i. Additionally, we will also have to remove all the pillars that are not part of the bridge, because they obstruct the river traffic. The cost of removing the i-th pillar is equal to w_i. This cost can even be negative—some interested parties are willing to pay you to get rid of certain pillars. All the heights h_i and costs w_i are integers.

What is the minimum possible cost of building the bridge that connects the first and last pillar?

Input

The first line contains the number of pillars, n. The second line contains pillar heights h_i in the order, separated by a space. The third line contains w_i in the same order, the costs of removing pillars.

Output

Output the minimum cost for building the bridge. Note that it can be negative.

Constraints

- $2 \leq n \leq 10^5$
- $0 \leq h_i \leq 10^6$
- $0 \leq |w_i| \leq 10^6$

Subtask 1 (30 points)

- $n \leq 1\,000$

Subtask 2 (30 points)

- optimal solution includes at most 2 additional pillars (besides the first and last)
- $|w_i| \leq 20$

Subtask 3 (40 points)

- no additional constraints

Example

Input

```
6
3 8 7 1 6 6
0 -1 9 1 2 0
```

Output

```
17
```

Palindromic Partitions

Time Limit: 10 s   *Memory Limit: 128 MB*

A *partition* of a string s is a set of one or more non-overlapping non-empty substrings of s (call them $a_1, a_2, a_3, \ldots, a_d$), such that s is their concatenation: $s = a_1 + a_2 + a_3 + \ldots + a_d$. We call these substrings *"chunks"* and define the *length* of such a partition to be the number of chunks, d.

We can represent the partition of a string by writing each chunk in parentheses. For example, the string `"decode"` can be partitioned as `(d)(ec)(ode)` or `(d)(e)(c)(od)(e)` or `(decod)(e)` or `(decode)` or `(de)(code)` or a number of other ways.

A partition is *palindromic* if its chunks form a palindrome when we consider each chunk as an atomic unit. For example, the only palindromic partitions of `"decode"` are `(de)(co)(de)` and `(decode)`. This also illustrates that every word has a trivial palindromic partition of length one.

Your task is to compute the maximal possible number of chunks in palindromic partition.

Input

The input starts with the number of test cases t in the first line. The following t lines describe individual test cases consisting of a single word s, containing only lowercase letters of the English alphabet. There are no spaces in the input.

Output

For every testcase output a single number: the length of the longest palindromic partition of the input word s.

Constraints

Let us denote the length of the input string s with n.

- $1 \le t \le 10$
- $1 \le n \le 10^6$

Subtask 1 (15 points)

- $n \le 30$

Subtask 2 (20 points)

- $n \le 300$

Subtask 3 (25 points)

- $n \le 10\,000$

Subtask 4 (40 points)

- no additional constraints

Example

Input

```
4
bonobo
deleted
racecar
racecars
```

Output

```
3
5
7
1
```

Chase

Time Limit: 4 s *Memory Limit: 512 MB*

Tom the cat is chasing Jerry the mouse again! Jerry is trying to gain some advantage by running into crowds of pigeons, where it is harder for Tom to follow him. Conveniently, Jerry arrived to Central Park in Ljubljana. The park has n statues, numbered $1 \dots n$, and $n-1$ non-intersecting passages connecting them in such a way that it is possible to reach any statue from any other statue by traversing the passages. Clustered tightly around each statue i there are p_i pigeons. Jerry has v breadcrumbs in his pocket. If he drops a breadcrumb by a statue at his current location, pigeons from neighbouring statues will immediately fly to this statue to eat the breadcrumb. As a result the current number of pigeons p around this and neighbouring statues changes.

It all happens in the following order: First, Jerry arrives to the statue i and meets p_i pigeons. Then, he drops the breadcrumb. He leaves the statue. The pigeons from neighbouring statues move to statue i before Jerry arrives to the next statue (so they don't count towards his count of the pigeons met).

Jerry may enter the park at any statue, run down some passages (but never using the same passage twice), and then leave the park anywhere he wants. After Jerry exits the park, Tom will enter and traverse exactly the same route. By dropping at most v breadcrumbs, Jerry wants to maximize the difference between the number of pigeons Tom will meet on the route and the number of pigeons he meets. Note that only pigeons that are present at some statue right before Jerry arrives there count toward the total number of pigeons he meets. See the comment of the example for further explanation.

Input

The first line contains the number of statues n and number of breadcrumbs v. The second line contains n integers separated with a space, $p_1 \dots p_n$. The following $n-1$ lines describe the passages with pairs of numbers a_i and b_i, which indicate there is a passage between statues a_i and b_i.

Output

Output only one number, maximum difference between the number of pigeons Tom meets and the number of pigeons Jerry meets.

Constraints

- $1 \leq n \leq 10^5$
- $0 \leq v \leq 100$
- $0 \leq p_i \leq 10^9$

Subtask 1 (20 points)

- $1 \leq n \leq 10$

Subtask 2 (20 points)

- $1 \leq n \leq 1000$

Subtask 3 (30 points)

- An optimal route begins at statue 1.

Subtask 4 (30 points)

- No additional constraints.

Example

Input

```
12 2
2 3 3 8 1 5 6 7 8 3 5 4
2 1
2 7
3 4
4 7
7 6
5 6
6 8
6 9
7 10
10 11
10 12
```

Output

```
36
```

Comment

One possible solution is the following. Jerry enters the park at the statue 6. There he encounters 5 pigeons. He drops a breadcrumb. p_6 is now 27 and $p_5 = p_7 = p_8 = p_9 = 0$. Next he runs to the statue 7 and encounters 0 pigeons. He drops the second breadcrumb. p_7 is now 41 and $p_2 = p_4 = p_6 = p_{10} = 0$. He exits the park. He encountered $5 + 0 = 5$ pigeons. Tom follows him over the same route but encounters $p_6 + p_7 = 0 + 41 = 41$ pigeons. The difference is $41 - 5 = 36$.