A. DZY Loves Physics

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

DZY loves Physics, and he enjoys calculating density.

Almost everything has density, even a graph. We define the density of a non-directed graph (nodes and edges of the graph have some values) as follows:

$$\begin{cases} \frac{v}{e} & (e > 0) \\ 0 & (e = 0) \end{cases}$$

where v is the sum of the values of the nodes, e is the sum of the values of the edges.
Once DZY got a graph G, now he wants to find a connected induced subgraph G' of the graph, such that the density of G' is as large as possible.

An induced subgraph $G'(V', E')$ of a graph $G(V, E)$ is a graph that satisfies:

- $V' \subseteq V$;
- edge $(a, b) \in E'$ if and only if $a \in V', b \in V'$, and edge $(a, b) \in E$;
- the value of an edge in G' is the same as the value of the corresponding edge in G, so as the value of a node.

Help DZY to find the induced subgraph with maximum density. Note that the induced subgraph you choose must be connected.

Input

The first line contains two space-separated integers n $(1 \le n \le 500)$, m $(0 \le m \le \frac{n(n-1)}{2})$. Integer n represents the number of nodes of the graph G, m represents the number of edges.

The second line contains n space-separated integers x_i $(1 \le x_i \le 10^6)$, where x_i represents the value of the i-th node. Consider the graph nodes are numbered from 1 to n.

Each of the next m lines contains three space-separated integers a_i, b_i, c_i $(1 \le a_i < b_i \le n;\ 1 \le c_i \le 10^3)$, denoting an edge between node a_i and b_i with value c_i. The graph won't contain multiple edges.

Output

Output a real number denoting the answer, with an absolute or relative error of at most 10^{-9}.

Examples

input

```
1 0
1
```

output

```
0.000000000000000
```

input

```
2 1
1 2
1 2 1
```

output

```
3.000000000000000
```

input

```
5 6
13 56 73 98 17
1 2 56
1 3 29
1 4 42
2 3 95
2 4 88
3 4 63
```

output

```
2.965517241379311
```

Note

In the first sample, you can only choose an empty subgraph, or the subgraph containing only node 1.

In the second sample, choosing the whole graph is optimal.

B. DZY Loves FFT

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

DZY loves Fast Fourier Transformation, and he enjoys using it.

Fast Fourier Transformation is an algorithm used to calculate convolution. Specifically, if a, b and c are sequences with length n, which are indexed from 0 to $n - 1$, and

$$c_i = \sum_{j=0}^{i} a_j b_{i-j}.$$

We can calculate c fast using Fast Fourier Transformation.

DZY made a little change on this formula. Now

$$c_i = \max_{j=0}^{i} a_j b_{i-j}.$$

To make things easier, a is a permutation of integers from 1 to n, and b is a sequence only containing 0 and 1. Given a and b, DZY needs your help to calculate c.

Because he is naughty, DZY provides a special way to get a and b. What you need is only three integers n, d, x. After getting them, use the code below to generate a and b.

```
//x is 64-bit variable;
function getNextX() {
    x = (x * 37 + 10007) % 1000000007;
    return x;
}
function initAB() {
    for(i = 0; i < n; i = i + 1){
        a[i] = i + 1;
    }
    for(i = 0; i < n; i = i + 1){
        swap(a[i], a[getNextX() % (i + 1)]);
    }
    for(i = 0; i < n; i = i + 1){
        if (i < d)
            b[i] = 1;
        else
            b[i] = 0;
    }
    for(i = 0; i < n; i = i + 1){
        swap(b[i], b[getNextX() % (i + 1)]);
    }
}
```

Operation `x % y` denotes remainder after division x by y. Function `swap(x, y)` swaps two values x and y.

Input

The only line of input contains three space-separated integers n, d, x ($1 \le d \le n \le 100000$; $0 \le x \le 1000000006$). Because DZY is naughty, x can't be equal to 27777500.

Output

Output n lines, the i-th line should contain an integer c_{i-1}.

Examples

input

```
3 1 1
```

output

```
1
3
2
```

input

```
5 4 2
```

output

```
2
2
4
5
5
```

input

```
5 4 3
```

output

```
5
5
5
5
4
```

Note

In the first sample, a is $[1\ 3\ 2]$, b is $[1\ 0\ 0]$, so $c_0 = max(1 \cdot 1) = 1$, $c_1 = max(1 \cdot 0, 3 \cdot 1) = 3$, $c_2 = max(1 \cdot 0, 3 \cdot 0, 2 \cdot 1) = 2$.

In the second sample, a is $[2\ 1\ 4\ 5\ 3]$, b is $[1\ 1\ 1\ 0\ 1]$.

In the third sample, a is $[5\ 2\ 1\ 4\ 3]$, b is $[1\ 1\ 1\ 1\ 0]$.

C. DZY Loves Colors

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

DZY loves colors, and he enjoys painting.

On a colorful day, DZY gets a colorful ribbon, which consists of n units (they are numbered from 1 to n from left to right). The color of the i-th unit of the ribbon is i at first. It is colorful enough, but we still consider that the colorfulness of each unit is 0 at first.

DZY loves painting, we know. He takes up a paintbrush with color x and uses it to draw a line on the ribbon. In such a case some contiguous units are painted. Imagine that the color of unit i currently is y. When it is painted by this paintbrush, the color of the unit becomes x, and the colorfulness of the unit increases by $|x - y|$.

DZY wants to perform m operations, each operation can be one of the following:

1. Paint all the units with numbers between l and r (both inclusive) with color x.
2. Ask the sum of colorfulness of the units between l and r (both inclusive).

Can you help DZY?

Input

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

Each of the next m lines begins with a integer $type$ $(1 \le type \le 2)$, which represents the type of this operation.

If $type = 1$, there will be 3 more integers l, r, x $(1 \le l \le r \le n;\ 1 \le x \le 10^8)$ in this line, describing an operation 1.

If $type = 2$, there will be 2 more integers l, r $(1 \le l \le r \le n)$ in this line, describing an operation 2.

Output

For each operation 2, print a line containing the answer — sum of colorfulness.

Examples

input

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

output

```
8
```

input

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

output

```
3
2
1
```

input

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

output

```
129
```

Note

In the first sample, the color of each unit is initially $[1, 2, 3]$, and the colorfulness is $[0, 0, 0]$.

After the first operation, colors become $[4, 4, 3]$, colorfulness become $[3, 2, 0]$.

After the second operation, colors become $[4, 5, 5]$, colorfulness become $[3, 3, 2]$.

So the answer to the only operation of type 2 is 8.

D. DZY Loves Strings

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

DZY loves strings, and he enjoys collecting them.

In China, many people like to use strings containing their names' initials, for example: xyz, jcvb, dzy, dyh.

Once DZY found a lucky string s. A lot of pairs of good friends came to DZY when they heard about the news. The first member of the i-th pair has name a_i, the second one has name b_i. Each pair wondered if there is a substring of the lucky string containing both of their names. If so, they want to find the one with minimum length, which can give them good luck and make their friendship last forever.

Please help DZY for each pair find the minimum length of the substring of s that contains both a_i and b_i, or point out that such substring doesn't exist.

A substring of s is a string $s_l s_{l+1} \dots s_r$ for some integers l, r $(1 \le l \le r \le |s|)$. The length of such the substring is $(r - l + 1)$.

A string p contains some another string q if there is a substring of p equal to q.

Input

The first line contains a string s $(1 \le |s| \le 50000)$.

The second line contains a non-negative integer q $(0 \le q \le 100000)$ — the number of pairs. Each of the next q lines describes a pair, the line contains two space-separated strings a_i and b_i $(1 \le |a_i|, |b_i| \le 4)$.

It is guaranteed that all the strings only consist of lowercase English letters.

Output

For each pair, print a line containing a single integer — the minimum length of the required substring. If there is no such substring, output -1.

Examples

input

```
xudyhduxyz
3
xyz xyz
dyh xyz
dzy xyz
```

output

```
3
8
-1
```

input

```
abcabd
3
a c
ab abc
ab d
```

output

```
2
3
3
```

input

```
baabcabaaa
2
abca baa
aa aba
```

output

```
6
4
```

Note

The shortest substrings in the first sample are: xyz, dyhduxyz.

The shortest substrings in the second sample are: ca, abc and abd.

The shortest substrings in the third sample are: baabca and abaa.

E. DZY Loves Planting

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

DZY loves planting, and he enjoys solving tree problems.

DZY has a weighted tree (connected undirected graph without cycles) containing n nodes (they are numbered from 1 to n). He defines the function $g(x, y)$ $(1 \le x, y \le n)$ as the longest edge in the shortest path between nodes x and y. Specially $g(z, z) = 0$ for every z.

For every integer sequence $p_1, p_2, \ldots, p_n$ $(1 \le p_i \le n)$, DZY defines $f(p)$ as $\min_{i=1}^{n} g(i, p_i)$.

DZY wants to find such a sequence p that $f(p)$ has maximum possible value. But there is one more restriction: the element j can appear in p at most x_j times.

Please, find the maximum possible $f(p)$ under the described restrictions.

Input

The first line contains an integer n $(1 \le n \le 3000)$.

Each of the next $n - 1$ lines contains three integers a_i, b_i, c_i $(1 \le a_i, b_i \le n;\ 1 \le c_i \le 10000)$, denoting an edge between a_i and b_i with length c_i. It is guaranteed that these edges form a tree.

Each of the next n lines describes an element of sequence x. The j-th line contains an integer x_j $(1 \le x_j \le n)$.

Output

Print a single integer representing the answer.

Examples

input

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

output

```
2
```

input

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

output

```
3
```

Note

In the first sample, one of the optimal p is $[4, 3, 2, 1]$.