Codeforces β
Sponsored by Telegram

Educational Codeforces Round 62 (Rated for Div. 2)

A. Detective Book

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

Ivan recently bought a detective book. The book is so interesting that each page of this book introduces some sort of a mystery, which will be explained later. The i-th page contains some mystery that will be explained on page a_i ($a_i \geq i$).

Ivan wants to read the whole book. Each day, he reads the first page he didn't read earlier, and continues to read the following pages one by one, until all the mysteries he read about are explained and clear to him (Ivan stops if there does not exist any page i such that Ivan already has read it, but hasn't read page a_i). After that, he closes the book and continues to read it on the following day from the next page.

How many days will it take to read the whole book?

Input

The first line contains single integer n ($1 \leq n \leq 10^4$) — the number of pages in the book.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($i \leq a_i \leq n$), where a_i is the number of page which contains the explanation of the mystery on page i.

Output

Print one integer — the number of days it will take to read the whole book.

Example

input

```
9
1 3 3 6 7 6 8 8 9
```

output

```
4
```

Note

Explanation of the example test:

During the first day Ivan will read only the first page. During the second day Ivan will read pages number 2 and 3. During the third day — pages 4-8. During the fourth (and the last) day Ivan will read remaining page number 9.

B. Good String

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

You have a string s of length n consisting of only characters > and <. You may do some operations with this string, for each operation you have to choose some character that still remains in the string. If you choose a character >, the character that comes right after it is deleted (if the character you chose was the last one, nothing happens). If you choose a character <, the character that comes right before it is deleted (if the character you chose was the first one, nothing happens).

For example, if we choose character > in string > **>** < >, the string will become to > > >. And if we choose character < in string > **<**, the string will become to <.

The string is good if there is a sequence of operations such that after performing it only one character will remain in the string. For example, the strings >, > > are good.

Before applying the operations, you may remove any number of characters from the given string (possibly none, possibly up to $n - 1$, but not the whole string). You need to calculate the minimum number of characters to be deleted from string s so that it becomes good.

Input

The first line contains one integer t ($1 \leq t \leq 100$) – the number of test cases. Each test case is represented by two lines.

The first line of i-th test case contains one integer n ($1 \leq n \leq 100$) – the length of string s.

The second line of i-th test case contains string s, consisting of only characters > and <.

Output
For each test case print one line.

For i-th test case print the minimum number of characters to be deleted from string s so that it becomes good.

Example

input
```
3
2
<>
3
><<
1
>
```
output
```
1
0
0
```

Note
In the first test case we can delete any character in string <>.

In the second test case we don't need to delete any characters. The string > < < is good, because we can perform the following sequence of operations: > **<** < $\rightarrow$ < **<** $\rightarrow$ <.

C. Playlist

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

You have a playlist consisting of n songs. The i-th song is characterized by two numbers t_i and b_i — its length and beauty respectively. The pleasure of listening to set of songs is equal to the total length of the songs in the set multiplied by the minimum beauty among them. For example, the pleasure of listening to a set of 3 songs having lengths $[5, 7, 4]$ and beauty values $[11, 14, 6]$ is equal to $(5 + 7 + 4) \cdot 6 = 96$.

You need to choose **at most** k songs from your playlist, so the pleasure of listening to the set of these songs them is maximum possible.

Input
The first line contains two integers n and k ($1 \le k \le n \le 3 \cdot 10^5$) – the number of songs in the playlist and the maximum number of songs you can choose, respectively.

Each of the next n lines contains two integers t_i and b_i ($1 \le t_i, b_i \le 10^6$) — the length and beauty of i-th song.

Output
Print one integer — the maximum pleasure you can get.

Examples

input
```
4 3
4 7
15 1
3 6
6 8
```
output
```
78
```

input
```
5 3
12 31
112 4
100 100
13 55
55 50
```
output
```
10000
```

Note
In the first test case we can choose songs $1, 3, 4$, so the total pleasure is $(4 + 3 + 6) \cdot 6 = 78$.

In the second test case we can choose song 3. The total pleasure will be equal to $100 \cdot 100 = 10000$.

D. Minimum Triangulation

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

You are given a regular polygon with n vertices labeled from 1 to n in counter-clockwise order. The triangulation of a given polygon is a set of triangles such that each vertex of each triangle is a vertex of the initial polygon, there is no pair of triangles such that their intersection has non-zero area, and the total area of all triangles is equal to the area of the given polygon. The weight of a triangulation is the sum of weigths of triangles it consists of, where the weight of a triagle is denoted as the product of labels of its vertices.

Calculate the minimum weight among all triangulations of the polygon.

Input

The first line contains single integer n ($3 \le n \le 500$) — the number of vertices in the regular polygon.

Output

Print one integer — the minimum weight among all triangulations of the given polygon.

Examples

input

```
3
```

output

```
6
```

input

```
4
```

output

```
18
```

Note

According to Wiki: polygon triangulation is the decomposition of a polygonal area (simple polygon) P into a set of triangles, i. e., finding a set of triangles with pairwise non-intersecting interiors whose union is P.

In the first example the polygon is a triangle, so we don't need to cut it further, so the answer is $1 \cdot 2 \cdot 3 = 6$.

In the second example the polygon is a rectangle, so it should be divided into two triangles. It's optimal to cut it using diagonal $1 - 3$ so answer is $1 \cdot 2 \cdot 3 + 1 \cdot 3 \cdot 4 = 6 + 12 = 18$.

E. Palindrome-less Arrays

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

Let's denote that some array b is *bad* if it contains a subarray $b_l, b_{l+1}, \dots, b_r$ of odd length more than 1 ($l < r$ and $r - l + 1$ is odd) such that $\forall i \in \{0, 1, \dots, r - l\}$ $b_{l+i} = b_{r-i}$.

If an array is not bad, it is **good**.

Now you are given an array $a_1, a_2, \dots, a_n$. Some elements are replaced by -1. Calculate the number of good arrays you can obtain by replacing each -1 with some integer from 1 to k.

Since the answer can be large, print it modulo 998244353.

Input

The first line contains two integers n and k ($2 \le n, k \le 2 \cdot 10^5$) — the length of array a and the size of "alphabet", i. e., the upper bound on the numbers you may use to replace -1.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($a_i = -1$ or $1 \le a_i \le k$) — the array a.

Output

Print one integer — the number of good arrays you can get, modulo 998244353.

Examples

input

```
2 3
-1 -1
```

output

```
9
```

input

```
5 2
1 -1 -1 1 2
```

output

```
0
```

input

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

output

```
2
```

input

```
4 200000
-1 -1 12345 -1
```

output

```
735945883
```

F. Extending Set of Points

time limit per test: 3.5 seconds
memory limit per test: 1024 megabytes
input: standard input
output: standard output

For a given set of two-dimensional points S, let's denote its extension $E(S)$ as the result of the following algorithm:

Create another set of two-dimensional points R, which is initially equal to S. Then, while there exist four numbers x_1, y_1, x_2 and y_2 such that $(x_1, y_1) \in R$, $(x_1, y_2) \in R$, $(x_2, y_1) \in R$ and $(x_2, y_2) \notin R$, add (x_2, y_2) to R. When it is impossible to find such four integers, let R be the result of the algorithm.

Now for the problem itself. You are given a set of two-dimensional points S, which is initially empty. You have to process two types of queries: add some point to S, or remove some point from it. After each query you have to compute the size of $E(S)$.

Input

The first line contains one integer q ($1 \le q \le 3 \cdot 10^5$) — the number of queries.

Then q lines follow, each containing two integers x_i, y_i ($1 \le x_i, y_i \le 3 \cdot 10^5$), denoting i-th query as follows: if $(x_i, y_i) \in S$, erase it from S, otherwise insert (x_i, y_i) into S.

Output

Print q integers. i-th integer should be equal to the size of $E(S)$ after processing first i queries.

Example

input

```
7
1 1
1 2
2 1
2 2
1 2
1 3
2 1
```

output

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

G. Double Tree

time limit per test: 10 seconds
memory limit per test: 1024 megabytes
input: standard input
output: standard output

You are given a special undirected graph. It consists of $2n$ vertices numbered from 1 to $2n$. The following properties hold for the graph:

- there are exactly $3n - 2$ edges in the graph: n edges connect vertices having odd numbers with vertices having even numbers, $n - 1$ edges connect vertices having odd numbers with each other, and $n - 1$ edges connect vertices having even numbers with each other;
- for each edge (u, v) between a pair of vertices with odd numbers, there exists an edge $(u + 1, v + 1)$, and vice versa;

- for each odd number $u \in [1, 2n-1]$, there exists an edge $(u, u+1)$;
- the graph is connected; moreover, if we delete all vertices with even numbers from it, and all edges incident to them, the graph will become a tree (the same applies to deleting odd vertices).

So, the graph can be represented as two trees having the same structure, and n edges connecting each vertex of the first tree to the corresponding vertex of the second tree.

Edges of the graph are weighted. The length of some simple path in the graph is the sum of weights of traversed edges.

You are given q queries to this graph; in each query, you are asked to compute the length of the shortest path between some pair of vertices in this graph. Can you answer all of the queries?

Input

The first line of the input contains one integer n ($2 \le n \le 3 \cdot 10^5$).

The second line contains n integers $w_{1,2}$, $w_{3,4}$, ..., $w_{2n-1,2n}$ ($1 \le w_{i,i+1} \le 10^{12}$). These integers describe the weights of the edges connecting odd vertices with even ones.

Then $n-1$ lines follow. i-th line contains four integers x_i, y_i, $w_{i,1}$ and $w_{i,2}$ ($1 \le x_i, y_i \le n$, $x_i \ne y_i$, $1 \le w_{i,j} \le 10^{12}$); it describes two edges: one connecting $2x_i - 1$ with $2y_i - 1$ and having weight $w_{i,1}$; another connecting $2x_i$ with $2y_i$ and having weight $w_{i,2}$.

The next line contains one integer q ($1 \le q \le 6 \cdot 10^5$) — the number of queries.

Then q lines follow, i-th line contains two integers u_i and v_i ($1 \le u_i, v_i \le 2n$, $u_i \ne v_i$), describing a query "compute the length of the shortest path between vertices u_i and v_i".

Output

Print q integers, i-th integer should be equal to the answer to the i-th query.

Example

input

```
5
3 6 15 4 8
1 2 5 4
2 3 5 7
1 4 1 5
1 5 2 1
3
1 2
5 6
1 10
```

output

```
3
15
4
```

Note

The graph in the first test looks like that: