Lista AA 2022 #7

A. Amr and Chemistry

1 second, 256 megabytes

Amr loves Chemistry, and specially doing experiments. He is preparing for a new interesting experiment.

Amr has n different types of chemicals. Each chemical i has an initial volume of a_i liters. For this experiment, Amr has to mix all the chemicals together, but all the chemicals volumes must be equal first. So his task is to make all the chemicals volumes equal.

To do this, Amr can do two different kind of operations.

- Choose some chemical i and double its current volume so the new volume will be $2a_i$
- Choose some chemical i and divide its volume by two (integer division) so the new volume will be $\lfloor \frac{a_i}{2} \rfloor$

Suppose that each chemical is contained in a vessel of infinite volume. Now Amr wonders what is the minimum number of operations required to make all the chemicals volumes equal?

Input

The first line contains one number n ($1 \le n \le 10^5$), the number of chemicals.

The second line contains n space separated integers a_i ($1 \le a_i \le 10^5$), representing the initial volume of the i-th chemical in liters.

Output

Output one integer the minimum number of operations required to make all the chemicals volumes equal.

input
3 4 8 2
output
2

input
3 3 5 6
output
5

In the first sample test, the optimal solution is to divide the second chemical volume by two, and multiply the third chemical volume by two to make all the volumes equal 4.

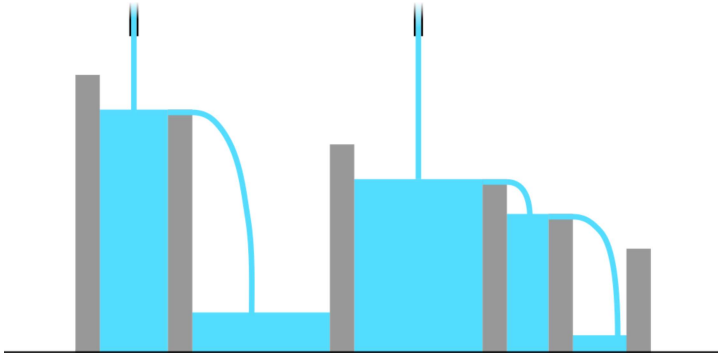
In the second sample test, the optimal solution is to divide the first chemical volume by two, and divide the second and the third chemical volumes by two twice to make all the volumes equal 1.

B. River Locks

2 seconds, 256 megabytes

Recently in Divanovo, a huge river locks system was built. There are now n locks, the i-th of them has the volume of v_i liters, so that it can contain any amount of water between 0 and v_i liters. Each lock has a pipe attached to it. When the pipe is open, 1 liter of water enters the lock every second.

The locks system is built in a way to immediately transfer all water exceeding the volume of the lock i to the lock $i + 1$. If the lock $i + 1$ is also full, water will be transferred further. Water exceeding the volume of the last lock pours out to the river.



The picture illustrates 5 locks with two open pipes at locks 1 and 3. Because locks 1, 3, and 4 are already filled, effectively the water goes to locks 2 and 5.

Note that the volume of the i-th lock may be greater than the volume of the $i + 1$-th lock.

To make all locks work, you need to completely fill each one of them. The mayor of Divanovo is interested in q independent queries. For each query, suppose that initially all locks are empty and all pipes are closed. Then, some pipes are opened simultaneously. For the j-th query the mayor asks you to calculate the minimum number of pipes to open so that all locks are filled no later than after t_j seconds.

Please help the mayor to solve this tricky problem and answer his queries.

Input

The first lines contains one integer n ($1 \le n \le 200\,000$) — the number of locks.

The second lines contains n integers $v_1, v_2, \ldots, v_n$ ($1 \le v_i \le 10^9$)) — volumes of the locks.

The third line contains one integer q ($1 \le q \le 200\,000$) — the number of queries.

Each of the next q lines contains one integer t_j ($1 \le t_j \le 10^9$) — the number of seconds you have to fill all the locks in the query j.

Output

Print q integers. The j-th of them should be equal to the minimum number of pipes to turn on so that after t_j seconds all of the locks are filled. If it is impossible to fill all of the locks in given time, print -1.

input
5 4 1 5 4 1 6 1 6 2 3 4 5

output

```
-1
3
-1
-1
4
3
```

input

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

output

```
-1
-1
4
4
-1
5
```

There are 6 queries in the first example test.

In the queries $1, 3, 4$ the answer is -1. We need to wait 4 seconds to fill the first lock even if we open all the pipes.

In the sixth query we can open pipes in locks 1, 3, and 4. After 4 seconds the locks 1 and 4 are full. In the following 1 second 1 liter of water is transferred to the locks 2 and 5. The lock 3 is filled by its own pipe.

Similarly, in the second query one can open pipes in locks 1, 3, and 4.

In the fifth query one can open pipes $1, 2, 3, 4$.

C. Kilani and the Game

2 seconds, 256 megabytes

Kilani is playing a game with his friends. This game can be represented as a grid of size $n \times m$, where each cell is either empty or blocked, and every player has one or more castles in some cells (there are no two castles in one cell).

The game is played in rounds. In each round players expand turn by turn: firstly, the first player expands, then the second player expands and so on. The expansion happens as follows: for each castle the player owns now, he tries to expand into the empty cells nearby. The player i can expand from a cell with his castle to the empty cell if it's possible to reach it in at most s_i (where s_i is player's expansion speed) moves to the left, up, right or down without going through blocked cells or cells occupied by some other player's castle. The player examines the set of cells he can expand to and builds a castle in each of them at once. The turned is passed to the next player after that.

The game ends when no player can make a move. You are given the game field and speed of the expansion for each player. Kilani wants to know for each player how many cells he will control (have a castle their) after the game ends.

Input

The first line contains three integers n, m and p ($1 \le n, m \le 1000$, $1 \le p \le 9$) — the size of the grid and the number of players.

The second line contains p integers s_i ($1 \le s \le 10^9$) — the speed of the expansion for every player.

The following n lines describe the game grid. Each of them consists of m symbols, where '.' denotes an empty cell, '#' denotes a blocked cell and digit x ($1 \le x \le p$) denotes the castle owned by player x.

It is guaranteed, that each player has at least one castle on the grid.

Output

Print p integers — the number of cells controlled by each player after the game ends.

input

```
3 3 2
1 1
1..
...
..2
```

output

```
6 3
```

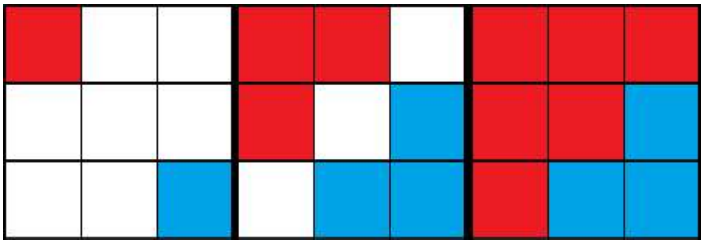
input

```
3 4 4
1 1 1 1
....
#...
1234
```

output

```
1 4 3 3
```

The picture below show the game before it started, the game after the first round and game after the second round in the first example:



In the second example, the first player is "blocked" so he will not capture new cells for the entire game. All other player will expand up during the first two rounds and in the third round only the second player will move to the left.

D. Dima and Salad

1 second, 256 megabytes

Dima, Inna and Seryozha have gathered in a room. That's right, someone's got to go. To cheer Seryozha up and inspire him to have a walk, Inna decided to cook something.

Dima and Seryozha have n fruits in the fridge. Each fruit has two parameters: the taste and the number of calories. Inna decided to make a fruit salad, so she wants to take some fruits from the fridge for it. Inna follows a certain principle as she chooses the fruits: the total taste to the total calories ratio of the chosen fruits must equal k. In other words, $\frac{\sum_{j=1}^{m} a_j}{\sum_{j=1}^{m} b_j} = k$, where a_j is the taste of the j-th chosen fruit and b_j is its calories.

Inna hasn't chosen the fruits yet, she is thinking: what is the maximum taste of the chosen fruits if she strictly follows her principle? Help Inna solve this culinary problem — now the happiness of a young couple is in your hands!

Inna loves Dima very much so she wants to make the salad from at least one fruit.

Input
The first line of the input contains two integers n, k ($1 \le n \le 100, 1 \le k \le 10$). The second line of the input contains n integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 100$) — the fruits' tastes. The third line of the input contains n integers $b_1, b_2, ..., b_n$ ($1 \le b_i \le 100$) — the fruits' calories. Fruit number i has taste a_i and calories b_i.

Output
If there is no way Inna can choose the fruits for the salad, print in the single line number -1. Otherwise, print a single integer — the maximum possible sum of the taste values of the chosen fruits.

input
3 2 10 8 1 2 7 1
output
18

input
5 3 4 4 4 4 4 2 2 2 2 2
output
-1

In the first test sample we can get the total taste of the fruits equal to 18 if we choose fruit number 1 and fruit number 2, then the total calories will equal 9. The condition $\frac{18}{9} = 2 = k$ fulfills, that's exactly what Inna wants.

In the second test sample we cannot choose the fruits so as to follow Inna's principle.

E. Median String

2 seconds, 256 megabytes

You are given two strings s and t, both consisting of exactly k lowercase Latin letters, s is lexicographically less than t.

Let's consider list of all strings consisting of exactly k lowercase Latin letters, lexicographically not less than s and not greater than t (including s and t) in lexicographical order. For example, for $k = 2$, $s =$"az" and $t =$"bf" the list will be ["az", "ba", "bb", "bc", "bd", "be", "bf"].

Your task is to print the median (the middle element) of this list. For the example above this will be "bc".

It is guaranteed that there is an odd number of strings lexicographically not less than s and not greater than t.

Input
The first line of the input contains one integer k ($1 \le k \le 2 \cdot 10^5$) — the length of strings.

The second line of the input contains one string s consisting of exactly k lowercase Latin letters.

The third line of the input contains one string t consisting of exactly k lowercase Latin letters.

It is guaranteed that s is lexicographically less than t.

It is guaranteed that there is an odd number of strings lexicographically not less than s and not greater than t.

Output
Print one string consisting exactly of k lowercase Latin letters — the median (the middle element) of list of strings of length k lexicographically not less than s and not greater than t.

input
2 az bf
output
bc

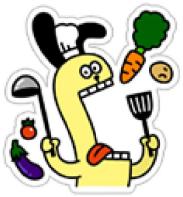
input
5 afogk asdji
output
alvuw

input
6 nijfvj tvqhwp
output
qoztvz

F. Tavas and Karafs

2 seconds, 256 megabytes

Karafs is some kind of vegetable in shape of an $1 \times h$ rectangle. Tavaspolis people love Karafs and they use Karafs in almost any kind of food. Tavas, himself, is crazy about Karafs.



Each Karafs has a positive integer height. Tavas has an infinite **1-based** sequence of Karafses. The height of the i-th Karafs is $s_i = A + (i - 1) \times B$.

For a given m, let's define an m-bite operation as decreasing the height of at most m distinct not eaten Karafses by 1. Karafs is considered as eaten when its height becomes zero.

Now SaDDas asks you n queries. In each query he gives you numbers l, t and m and you should find the largest number r such that $l \le r$ and sequence $s_l, s_{l+1}, ..., s_r$ can be eaten **by performing *m*-bite no more than *t* times** or print -1 if there is no such number r.

Input
The first line of input contains three integers A, B and n ($1 \le A, B \le 10^6$, $1 \le n \le 10^5$).

Next n lines contain information about queries. i-th line contains integers l, t, m ($1 \le l, t, m \le 10^6$) for i-th query.

Output
For each query, print its answer in a single line.

input

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

output

```
4
-1
8
-1
```

input

```
1 5 2
1 5 10
2 7 4
```

output

```
1
2
```

G. GCD Counting

4.5 seconds, 256 megabytes

You are given a tree consisting of n vertices. A number is written on each vertex; the number on vertex i is equal to a_i.

Let's denote the function $g(x, y)$ as the greatest common divisor of the numbers written on the vertices belonging to the simple path from vertex x to vertex y (including these two vertices). Also let's denote $dist(x, y)$ as the number of vertices on the simple path between vertices x and y, including the endpoints. $dist(x, x) = 1$ for every vertex x.

Your task is calculate the maximum value of $dist(x, y)$ among such pairs of vertices that $g(x, y) > 1$.

Input

The first line contains one integer n — the number of vertices $(1 \le n \le 2 \cdot 10^5)$.

The second line contains n integers a_1, a_2, ..., a_n $(1 \le a_i \le 2 \cdot 10^5)$ — the numbers written on vertices.

Then $n - 1$ lines follow, each containing two integers x and y $(1 \le x, y \le n, x \ne y)$ denoting an edge connecting vertex x with vertex y. It is guaranteed that these edges form a tree.

Output

If there is no pair of vertices x, y such that $g(x, y) > 1$, print 0. Otherwise print the maximum value of $dist(x, y)$ among such pairs.

input

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

output

```
1
```

input

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

output

```
2
```

input

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

output

```
0
```

H. Fire

2 seconds, 256 megabytes

Polycarp is in really serious trouble — his house is on fire! It's time to save the most valuable items. Polycarp estimated that it would take t_i seconds to save i-th item. In addition, for each item, he estimated the value of d_i — the moment after which the item i will be completely burned and will no longer be valuable for him at all. In particular, if $t_i \ge d_i$, then i-th item cannot be saved.

Given the values p_i for each of the items, find a set of items that Polycarp can save such that the total value of this items is maximum possible. Polycarp saves the items one after another. For example, if he takes item a first, and then item b, then the item a will be saved in t_a seconds, and the item b — in $t_a + t_b$ seconds after fire started.

Input

The first line contains a single integer n $(1 \le n \le 100)$ — the number of items in Polycarp's house.

Each of the following n lines contains three integers t_i, d_i, p_i $(1 \le t_i \le 20, 1 \le d_i \le 2\,000, 1 \le p_i \le 20)$ — the time needed to save the item i, the time after which the item i will burn completely and the value of item i.

Output

In the first line print the maximum possible total value of the set of saved items. In the second line print one integer m — the number of items in the desired set. In the third line print m distinct integers — numbers of the saved items *in the order Polycarp saves them*. Items are 1-indexed in the same order in which they appear in the input. If there are several answers, print any of them.

input

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

output

```
11
2
2 3
```

input

```
2
5 6 1
3 3 5
```

output

```
1
1
1
```

In the first example Polycarp will have time to save any two items, but in order to maximize the total value of the saved items, he must save the second and the third item. For example, he can firstly save the third item in 3 seconds, and then save the second item in another 2 seconds. Thus, the total value of the saved items will be $6 + 5 = 11$.

In the second example Polycarp can save only the first item, since even if he immediately starts saving the second item, he can save it in 3 seconds, but this item will already be completely burned by this time.

I. Nauuo and Circle

2 seconds, 256 megabytes

Nauuo is a girl who loves drawing circles.

One day she has drawn a circle and wanted to draw a tree on it.

The tree is a connected undirected graph consisting of n nodes and $n - 1$ edges. The nodes are numbered from 1 to n.

Nauuo wants to draw a tree on the circle, the nodes of the tree should be in n **distinct** points on the circle, and the edges should be straight without crossing each other.

"Without crossing each other" means that every two edges have no common point or the only common point is an endpoint of both edges.

Nauuo wants to draw the tree using a permutation of n elements. A permutation of n elements is a sequence of integers $p_1, p_2, \ldots, p_n$ in which every integer from 1 to n appears exactly once.

After a permutation is chosen Nauuo draws the i-th node in the p_i-th point on the circle, then draws the edges connecting the nodes.

The tree is given, Nauuo wants to know how many permutations are there so that the tree drawn satisfies the rule (the edges are straight without crossing each other). She only wants to know the answer modulo 998244353, can you help her?

It is obvious that whether a permutation is valid or not does not depend on which n points on the circle are chosen.

Input

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

Each of the next $n - 1$ lines contains two integers u and v ($1 \le u, v \le n$), denoting there is an edge between u and v.

It is guaranteed that the given edges form a tree.

Output

The output contains a single integer — the number of permutations suitable to draw the given tree on a circle satisfying the rule, modulo 998244353.

input

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

output

```
16
```

input

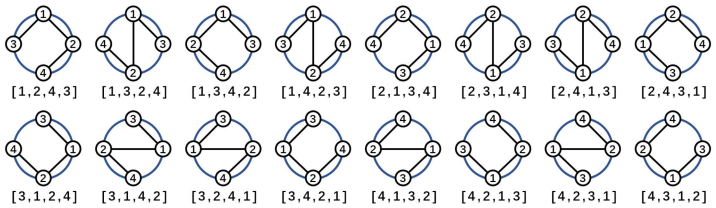
```
4
1 2
1 3
1 4
```

output

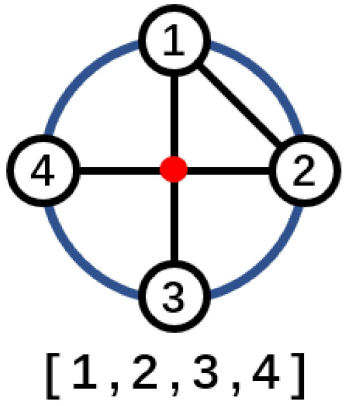
```
24
```

Example 1

All valid permutations and their spanning trees are as follows.



Here is an example of invalid permutation: the edges $(1, 3)$ and $(2, 4)$ are crossed.



Example 2

Every permutation leads to a valid tree, so the answer is $4! = 24$.

J. Two Paths

2 seconds, 64 megabytes

As you know, Bob's brother lives in Flatland. In Flatland there are n cities, connected by n - 1 two-way roads. The cities are numbered from 1 to n. You can get from one city to another moving along the roads.

The «Two Paths» company, where Bob's brother works, has won a tender to repair two paths in Flatland. A path is a sequence of different cities, connected sequentially by roads. The company is allowed to choose by itself the paths to repair. The only condition they have to meet is that the two paths shouldn't cross (i.e. shouldn't have common cities).

It is known that the profit, the «Two Paths» company will get, equals the product of the lengths of the two paths. Let's consider the length of each road equals 1, and the length of a path equals the amount of roads in it. Find the maximum possible profit for the company.

Input

The first line contains an integer n ($2 \le n \le 200$), where n is the amount of cities in the country. The following n - 1 lines contain the information about the roads. Each line contains a pair of numbers of the cities, connected by the road a_i, b_i ($1 \le a_i, b_i \le n$).

Output

Output the maximum possible profit.

input

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

output

```
1
```

input

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

output

```
0
```

input

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

output

```
4
```