Problem A. Albuquerque's Avenues

Time limit: 3 seconds
Memory limit: 256 megabytes

Albuquerque is a large city. It has n crossings and $n-1$ one-directional roads between them. It's guaranteed that if all roads were undirected, it would be possible to get from any crossing to any other by these roads. Additionally, every crossing has a label: `S` or `F`, indicating start and finish correspondingly.

Saul and Kim are going to do another chicanery. They can perform the following operation any number of times:

- Choose any road (u, v) such that the following conditions hold:
  - This road is directed from crossing u to crossing v
  - Crossing u has label `S`
  - Crossing v has label `F`
- Change the direction of the road and swap labels on u and v.

What is the total number of different configurations that Saul and Kim can achieve by applying this operation any number of times? As this number can be very large, output it modulo 998244353.

Two configurations are called different if some crossing has different labels in them, or if some road has different orientation in them.

Input

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

The second line of the input contains a string of length n, consisting of characters `S` and `F`. The i-th character of this string denotes the initial label of crossing i.

The i-th of the next $n-1$ lines contains two integers u_i, v_i $(1 \le u_i, v_i \le n,\ u_i \ne v_i)$ — denoting a directed road from crossing u_i to crossing v_i. It's guaranteed that if all roads were undirected, it would be possible to get from any crossing to any other by these roads.

Output

Output a single integer — the total number of different configurations that Saul and Kim can achieve by applying this operation any number of times, modulo 998244353.

Examples

standard input	standard output
5 SFSFS 2 1 2 3 4 3 4 5	1
4 SFFF 1 2 1 3 1 4	4
7 SSSSFFF 1 2 3 2 4 3 4 5 4 6 2 7	13

Note

In the first sample, all roads are directed from crossings with `F` on them to crossings with `S` on them, so it's impossible to do any operations.

In the second sample, for each $v = 2, 3, 4$ it's possible to do an operation with nodes $1, v$. Resulting three configurations, together with the initial one, are the only possible configurations.

Problem B. Breaking Bad

Time limit: 1 second
Memory limit: 256 megabytes

Walter White finally broke bad and finished building his drug empire.

He has a network of n distributors, which looks like a tree. Jesse is the head distributor, numbered 0. For every $i \geq 1$, i-th distributor has a direct supervisor — distributor p_i with $p_i < i$. Initially, distributor i has a_i dollars.

Walter is going to shout for k times. Every time Walter shouts, the following will happen:

- For all $1 \leq i < n$, i-th distributor will give all his money to its direct supervisor (all of them do this simultaneously).

After shouting k times, Walter will choose the distributor with the maximum amount of money and collect all his money.

For all k from 0 to $n-1$, find the amount of money Walter would collect after shouting exactly k times.

Input

The first line of input contains a single integer t ($1 \leq t \leq 2 \cdot 10^5$) — the number of test cases. The description of test cases follows.

The first line of each test case consists of a single integer n ($1 \leq n \leq 2 \cdot 10^5$) — the number of distributors.

The second line of each test case consists of $n-1$ integers $p_1, p_2, \ldots, p_{n-1}$ ($0 \leq p_i < i$) — where p_i is a direct supervisor of distributor i.

The third line of each test case consists of n integers $a_0, a_1, \ldots, a_{n-1}$ ($0 \leq a_i \leq 10^9$) — the initial amounts of money that distributors have.

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

Output

For each test case print n integers — the amount of money Walter would collect after shouting exactly k times, for all k from 0 to $n-1$.

Example

standard input	standard output

3
4
0 1 1
1 3 4 2
5
0 1 2 1
10 78 19 36 64
6
0 1 2 2 4
25 29 27 20 23 28
	4 6 10 10
	78 88 171 207 207
	29 54 81 124 152 152

Note

Consider the first test case.

- If Walt shouts 0 times, the amounts of money of distributors are $(1, 3, 4, 2)$. So, he will collect 4 dollars.
- If Walt shouts 1 time, the amounts of money of distributors are $(4, 6, 0, 0)$. So, he will collect 6 dollars.
- If Walt shouts 2 times, the amounts of money of distributors are $(10, 0, 0, 0)$. So, he will collect 10 dollars.
- If Walt shouts 3 times, the amounts of money of distributors are $(10, 0, 0, 0)$. So, he will collect 10 dollars.

Problem C. Chemistry Class

Time limit: 1 second
Memory limit: 256 megabytes

Walter White has $2n$ students in his chemistry class. Student i has chemistry skill a_i.

He wants to divide students into n pairs for a group exercise. The pair works better together if their skills are closer. More precisely,

- If skills of students in a pair differ by more than A, they will **blow up the lab**;
- If skills of students in a pair differ by at most A, but by more than B, they will **produce a mediocre product**;
- If skills of students differ by at most B, they will **produce** 99.1% **pure product**.

Walter wants to split students into n pairs so that:

- Lab is not blown up;
- The number of pairs that produced 99.1% pure product is as large as possible.

Determine if Walter can split students in such a way, and if he can, find the largest possible number of pairs that would produce 99.1% pure product.

Input

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

The first line of each test case contains three integers n, A, B ($1 \le n \le 2 \cdot 10^5, 1 \le B < A \le 10^{18}$) — the number of students.

The second line of each test case contains $2n$ integers $a_1, a_2, \ldots, a_{2n}$ ($0 \le a_i \le 10^{18}$) — skills of the students.

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

Output

For each test case, if there is no way to split the students into pairs without blowing up the lab, output -1. Otherwise, output the largest possible number of pairs that would produce 99.1% pure product.

Example

standard input	standard output

4
1 2 1
42 69
2 3 1
1 2 3 4
2 5 1
6 1 3 4
5 19 1
1 7 8 9 10 11 12 13 14 20
	-1
	2
	1
	4

Note

In the first test case, it's impossible to split students into pairs without blowing up the lab.

In the second test case, we can pair the first student with the second, and the third one with the fourth one. Both pairs will have a difference in skills equal to 1, and both will produce 99.1% pure product.

Problem D. Daily Disinfection

Time limit: 1 second
Memory limit: 256 megabytes

To get 99.1% pure product, everything in the lab has to be thoroughly cleaned daily. Right now, Jesse is going to clean a shelf with books.

The shelf consists of n places for books. There are some books placed on the shelf (possibly, none). If some place is empty, Jesse can clean it without any power consumption (he is really good at cleaning). He cannot clean any place with a book currently in it. If there is a book at some place, and the adjacent place is empty, Jesse can move the book to that adjacent place consuming 1 power.

Jesse is a very busy person, and he does not want to spend too much power. For each test case, tell the minimal amount of power Jesse has to spend.

Input

The first line contains of the single integer t $(1 \le t \le 10^5)$ — the number of the test cases.

The first line of each of the t test cases contains a single integer n $(1 \le n \le 2 \cdot 10^5)$ — the number of the places on the shelf.

The second line contains a string s of length n, where $s_i = 0$ if there is no book at i-th place, and $s_1 = 1$ otherwise.

It is guaranteed that in all provided test cases it is possible to clean the shelf.

The sum of n over all test cases will not be greater than $2 \cdot 10^5$.

Output

Output t lines, each line containing a single integer — the minimal amount of power Jesse has to spend.

Example

standard input	standard output
3 2 01 5 00110 9 101010101	1 2 6

Note

In the first test case, Jesse can do the following:

- Clean the first place;
- Move the book from the second place to the first (consuming 1 power);
- Clean the second place.

Problem E. Equalizer Ehrmantraut

Time limit: 1 second
Memory limit: 256 megabytes

When Mike Ehrmantraut started being a cop, he wanted all people to be equal before the law. Now he is retired, but he still wants all elements of an integer array a of length n to become equal.

He can perform the following operations:

1. Choose any i such that $1 \le i \le n-1$ and set $a_i = a_i \texttt{ AND } a_{i+1}$.
2. Choose any i such that $2 \le i \le n$ and set $a_i = a_i \texttt{ AND } a_{i-1}$.

Find the smallest number of operations Mike needs to do to make all elements of a equal.

Here `AND` denotes the bitwise AND operation.

Input

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

The first line of each test case contains a single integer n ($2 \le n \le 2 \cdot 10^5$) — the length of the array.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i < 2^{30}$) — the elements of the array.

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 a single integer — the answer to the problem.

Example

standard input	standard output
3	0
2	5
2 2	1
5	
1 2 3 4 5	
4	
2 2 3 2	

Note

In the first test case, all numbers are already equal, so Mike doesn't need to do any operations.

In the second test case, it's enough to do the following 5 operations:

- Set a_2 equal to $a_1 \texttt{ AND } a_2 = 1 \texttt{ AND } 2 = 0$.
- Set a_1 equal to $a_1 \texttt{ AND } a_2 = 1 \texttt{ AND } 0 = 0$.
- Set a_3 equal to $a_2 \texttt{ AND } a_3 = 0 \texttt{ AND } 3 = 0$.
- Set a_4 equal to $a_3 \texttt{ AND } a_4 = 0 \texttt{ AND } 4 = 0$.
- Set a_5 equal to $a_4 \texttt{ AND } a_5 = 0 \texttt{ AND } 5 = 0$.

Problem F. Felina

Time limit: 1 second
Memory limit: 256 megabytes

Help Walter solve his final problem.

For a positive integer n, define $highest\_bit(n)$ as the largest number i, such that $2^i \le n$. Also define $highest\_bit(0) = -1$.

You are given a positive integer X. Find the number of multisets S of positive integers, which satisfy the following conditions:

- All elements of S are nonnegative powers of 2.
- The sum of elements of S is X.
- There is no way to split elements of S into two groups so that $highest\_bit(S_1) = highest\_bit(S_2)$, where S_1 is the sum of the elements in the first group, and S_2 is the sum of the elements in the second group.

Solve this problem for $X = 1, 2, \ldots, n$.

Since the answers can be very large, output them modulo 998244353.

Input

The only line of the input contains a single integer n ($1 \le n \le 10^6$).

Output

Output n integers: answers to the problem for $X = 1, 2, \ldots, n$, modulo 998244353.

Example

standard input	standard output
10	1 1 2 1 1 3 6 1 1 2

Problem G. Genius Gus

Time limit: 1 second
Memory limit: 256 megabytes

Gus is given a binary string s. He has to perform the following operation **exactly once**:

- Choose some subsequence of the string. Then, arrange characters on those positions in the sorted order.

For example, for string `01`$\underline{\texttt{10}}$`1`$\underline{\texttt{01}}$`1`$\underline{\texttt{0}}$ Gus can choose symbols at positions $3, 4, 6, 7, 9$ (underlined). Symbols at these positions form string `10010`. After sorting them, Gus will get the string `01`$\underline{\texttt{00}}$`1`$\underline{\texttt{01}}$`1`$\underline{\texttt{1}}$.

Gus wants to find the number of possible strings he might get after applying this operation **exactly once**. For him this problem is trivial, but can you solve it?

As this number can be very large, output it modulo 998244353.

Input

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 only line of each test case contains a binary string s.

It is guaranteed that the sum of lengths of s over all test cases does not exceed 10^6.

Output

For each test case, output a single integer — the number of possible strings Gus might get after applying this operation **exactly once**.

Example

standard input	standard output
6	1
0	1
1	1
01	2
10	4
1010	437686
0101010111001010100101011100101001010	

Note

For the fourth test case, the string is `10`. If he selects both positions, he will get string `01`. For any other selection, the string will remain `10`. So, there are two possible strings he might obtain.

For the fifth test case, here are all possible strings: `1010`, `0011`, `0110`, `1001`.

Problem H. Healthy Howard Hamlin

Time limit: 1 second
Memory limit: 256 megabytes

Howard Hamlin likes to run. He decided that he is going to run for A consecutive minutes today.

Howard can run in two modes: slow and fast. The following conditions hold:

- Howard can run in fast mode for at most B minutes in a row;
- After Howard ran in a fast mode for some time, he has to rest for at least the same amount of time. More formally, if Howard ran for X minutes in a row in fast mode and switched to the slow mode, he won't be able to run in fast mode for at least X next minutes;
- Howard can only switch running modes at integer points in time.

Howard wants to run in fast mode as much as possible. What's the largest number of minutes out of all A minutes that Howard could spend running in a fast mode?

Input

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

The only line of each test case contains two integers A, B $(1 \le B \le A \le 10^9)$.

Output

For each test case, output a single integer — the largest number of minutes out of all A minutes that Howard could spend running in a fast mode.

Example

standard input	standard output

3
3 1
10 10
69 42
	2
	10
	55

Note

In the first test case, Howard can run for the first minute in the fast mode, then run for the second minute in the slow mode, and then run for the third minute in fast mode again.

In the second test case, Howard can run for all 10 minutes in the fast mode.

Problem I. Immense Input

Time limit: 1 second
Memory limit: 256 megabytes

Once Charles McGill prepared a problem for SEERC. It's input section looked like this:

The first line contains a single integer t $(1 \le t \le T)$ — the number of test cases.

The first line of each test case contains a single positive integer n.

$\vdots$

It is guaranteed that the sum of n^2 over all test cases does not exceed K.

Chuck realized that he forgot to tell anything about the sum of n. What if it's too large and it might take too long to read the input?

Unfortunately, Chuck suffers from electromagnetic hypersensitivity, so he can't use his computer to check this. Help him: find the largest possible sum of n under the constraints from the statement.

Input

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

The only line of each test case contains two integers T, K $(1 \le T \le 10^6$, $1 \le K \le 10^{18})$.

Output

For each test case, output a single integer — the largest possible sum of n under the constraints of the problem.

Example

standard input	standard output

3
4 30
1 1000000000000000000
69 42
	10
	1000000000
	42

Note

In the first test case, the largest possible sum is 10. One way to get it is when $t = 4$ and corresponding values of n are $1, 2, 3, 4$: $1^2 + 2^2 + 3^2 + 4^2 = 30, 1 + 2 + 3 + 4 = 10$. Another way to get it is with the following values of n: $2, 2, 3, 3$: $2^2 + 2^2 + 3^2 + 3^2 = 26 < 30, 2 + 2 + 3 + 3 = 10$.

Problem J. Jesse's Job

Time limit: 2 seconds
Memory limit: 256 megabytes

Jesse has a permutation $p_1, p_2, \ldots, p_n$ of integers from 1 to n. His job is simple: to maximize the number of positions i, for which $p_i = i$. To achieve that, Jesse can do the following operation **exactly once**:

- Color some elements of the permutation in yellow, and all the remaining elements in blue. There has to be at least one yellow and at least one blue element.
- Then, separately sort yellow and blue numbers.

For example, for permutation $(3, 5, 1, 6, 2, 4)$, Jesse can mark numbers $3, 5, 4$ yellow, and $1, 6, 2$ blue. After sorting yellow and blue elements separately, Jesse will get permutation $(3, 4, 1, 2, 6, 5)$.

Jesse's score in the end is the number of positions i, for which $p_i = i$. Find the maximal score Jesse can achieve and some way to achieve it.

Input

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

The first line of each test case contains a single integer n $(2 \le n \le 10^6)$ — the length of the permutation.

The second line of each test case contains n integers $p_1, p_2, \ldots, p_n$ $(1 \le p_i \le n$, all p_i are distinct$)$ — elements of the permutation.

It is guaranteed that the sum of n over all test cases does not exceed 10^6.

Output

For every test case, in the first line, output the maximal score Jesse can achieve.

In the second line, output a single integer k $(1 \le k \le n-1)$ — the number of elements Jesse should color yellow.

In the third line, output k integers $pos_1, pos_2, \ldots, pos_k$ $(1 \le pos_i \le n$, all pos_i are distinct$)$ — the **positions** of elements that you are going to color in yellow.

If there are several ways to obtain the maximum score, output any of them.

Example

standard input	standard output

3
2
2 1
4
2 1 4 3
6
3 5 4 2 6 1
	0
	1
	1
	4
	2
	1 2
	4
	3
	1 3 4

Note

In the first test case, for permutation $(p_1, p_2) = (2, 1)$, Jesse can mark p_1 yellow and p_2 blue. After sorting it will remain being $(2, 1)$, with score 0.

In the second test case, for permutation $(p_1, p_2, p_3, p_4) = (2, 1, 4, 3)$, Jesse can mark p_1, p_2 yellow and p_3, p_4 blue. After sorting the permutation will become $(1, 2, 3, 4)$, with score 4.

Problem K. Kim

Time limit: 1 second
Memory limit: 256 megabytes

Kim Wexler is tired of working for Mesa Verde. Instead, she wants to solve the following problem.

Kim has two integers a and b. Help her to find any x such that $0 \le x \le 10^9$ and the value of $(a + x)$ `AND` $(b + x)$ is minimized.

Here `AND` denotes the bitwise AND operation.

Input

The first line contains a single integer t ($1 \le t \le 2 \cdot 10^5$) — the number of test cases. The description of test cases follows.

The only line of each test case contains two integers a and b ($1 \le a, b \le 10^9$).

Output

For each test case, output a single integer — the answer to the problem. If there are several integers x such that $0 \le x \le 10^9$ and the value of $(a + x)$ `AND` $(b + x)$ is minimized, you can output any of them.

Example

standard input	standard output

2
1 11
64 64
	1
	0

Note

In the first test case, Kim can select $x = 1$ to get $(1 + 1)$ `AND` $(11 + 1) = 2$ `AND` $12 = 0$.

In the second test case, Kim can select $x = 0$ to get $(64 + 0)$ `AND` $(64 + 0) = 64$ `AND` $64 = 64$.

Problem L. Lalo's Lawyer Lost

Time limit: 1 second
Memory limit: 256 megabytes

While picking up Lalo's bail, Saul and Mike got lost in a desert. In a desert, they found a cactus: a connected undirected graph on n nodes, such that every edge in it appears in at most one simple cycle.

It turned out that n is even. For some reason, Mike and Saul want to solve the following problem:

Define $d(u, v)$ as the shortest distance between nodes u and v in this cactus. Partition all n nodes into $\frac{n}{2}$ pairs (u_i, v_i), such that every node appears in exactly one pair, and the sum of $d(u_i, v_i)$ is **maximized**.

What's the maximum possible sum of $d(u_i, v_i)$ in such a partition?

Input

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

The first line of each test case contains two integers n, m $(2 \le n \le 2 \cdot 10^5, n-1 \le m \le 4 \cdot 10^5$, n **is even**) — the number of nodes and edges correspondingly.

The i-th of the next m lines contains two integers u_i, v_i $(1 \le u_i, v_i \le n,\ u_i \ne v_i)$ — denoting an edge between nodes u_i and v_i. It's guaranteed that these edges form a cactus.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$, and the sum of m over all test cases does not exceed $4 \cdot 10^5$.

Output

For each test case, output a single integer — the largest possible sum of $d(u_i, v_i)$ in such a partition.

Example

standard input	standard output
3	4
4 3	7
1 2	11
2 3	
3 4	
6 7	
1 2	
2 3	
3 1	
3 4	
4 5	
5 6	
6 4	
8 9	
1 2	
2 3	
3 1	
3 4	
4 5	
5 6	
6 7	
7 8	
8 3	