Kotlin Heroes: Episode 6

A. From Zero To Y

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

You are given two positive (greater than zero) integers x and y. There is a variable k initially set to 0.

You can perform the following two types of operations:

- add 1 to k (i. e. assign $k := k + 1$);
- add $x \cdot 10^p$ to k for some non-negative p (i. e. assign $k := k + x \cdot 10^p$ for some $p \geq 0$).

Find the minimum number of operations described above to set the value of k to y.

Input

The first line contains one integer t ($1 \leq t \leq 2 \cdot 10^4$) — the number of test cases.

Each test case consists of one line containing two integer x and y ($1 \leq x, y \leq 10^9$).

Output

For each test case, print one integer — the minimum number of operations to set the value of k to y.

Example

input
3 2 7 3 42 25 1337
output
4 5 20

Note

In the first test case you can use the following sequence of operations:

1. add 1;
2. add $2 \cdot 10^0 = 2$;
3. add $2 \cdot 10^0 = 2$;
4. add $2 \cdot 10^0 = 2$.

$1 + 2 + 2 + 2 = 7$.

In the second test case you can use the following sequence of operations:

1. add $3 \cdot 10^1 = 30$;
2. add $3 \cdot 10^0 = 3$;
3. add $3 \cdot 10^0 = 3$;
4. add $3 \cdot 10^0 = 3$;
5. add $3 \cdot 10^0 = 3$.

$30 + 3 + 3 + 3 + 3 = 42$.

B. RBS Deletion

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

A bracket sequence is a string containing only characters "(" and ")". A regular bracket sequence (or, shortly, an RBS) is a bracket sequence that can be transformed into a correct arithmetic expression by inserting characters "1" and "+" between the original characters of the sequence. For example:

- bracket sequences "()()" and "(())" are regular (the resulting expressions are: "(1)+(1)" and "((1+1)+1)");

- bracket sequences "()", "(" and ")" are not.

You are given a string s, which is an RBS. You can apply any number of operations to this string. Each operation can have one of the following types:

1. choose some non-empty prefix of s and remove it from s, so s is still an RBS. For example, we can apply this operation as follows: "**(()())(())**()()" $\rightarrow$ "()()" (the first 10 characters are removed);
2. choose some **contiguous** non-empty substring of s and remove it from s, so s is still an RBS. For example, we can apply this operation as follows: "(())()**(())**()()" $\rightarrow$ "(())()()()" (the characters from the 7-th to the 10-th are removed).

The operation 2 can be applied at most k times. Calculate the maximum number of operations you can apply until s becomes empty.

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

Each test case is described by two lines. The first line contains two integers n and k ($2 \le n \le 2 \cdot 10^5$; $1 \le k \le n$; n is even) — the length of s and the maximum number of operations of type 2 you can apply.

The second line contains a string s of n characters '(' and ')'. This string is an RBS.

The sum of n over all test cases doesn't exceed $2 \cdot 10^5$.

Output
For each test case, print one integer — the maximum number of operations you can apply.

Example

input
3 12 2 ((()())((()))) 6 3 ()()() 8 1 (((())))
output
4 3 2

C. Two Policemen

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

There is a street that can be represented as an array of length n.

There are two policemen patrolling a street: the first one is standing at the point x of the street and the second one is standing at the point y of the street.

During one minute, both policemen can decide what to do (independently): move left (if the current position is greater than 1), move right (if the current position is less than n), or do nothing.

The street is considered clear if **each** point of the street is visited by at least one policeman.

Your task is to find the minimum number of minutes the policemen need to visit **each** point of the street (again, each point should be visited by at least one of them).

You have to answer t independent test cases.

Input
The first line of the input contains one integer t ($1 \le t \le 2 \cdot 10^4$) — the number of test cases. Then t test cases follow.

The only line of the test case contains three integers n, x and y ($2 \le n \le 10^6$; $1 \le x, y \le n$; $x \ne y$) — the length of the street, the position of the first policeman and the position of the second policeman, respectively.

It is guaranteed that the sum of n does not exceed 10^6 ($\sum n \le 10^6$).

Output
For each test case, print one integer — the minimum number of minutes the policemen need to visit **each** point of the street.

Example

input
6 4 1 2

```
7 7 1
10 2 6
8 5 2
2 1 2
20 4 14
```

output

```
2
3
5
4
0
12
```

D. Problemsolving Marathon

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

Polycarp has decided to do a problemsolving marathon. He wants to solve s problems in n days. Let a_i be the number of problems he solves during the i-th day. He wants to find a distribution of problems into days such that:

- a_i is an integer value for all i from 1 to n;
- $a_i \geq 1$ for all i from 1 to n;
- $a_{i+1} \geq a_i$ for all i from 1 to $n-1$;
- $a_{i+1} \leq 2 \cdot a_i$ for all i from 1 to $n-1$;
- a_n is maximized.

Note that a_1 can be arbitrarily large.

What is the largest value of a_n Polycarp can obtain?

Input

The first line contains a single integer t ($1 \leq t \leq 1000$) — the number of testcases.

Then the descriptions of t testcases follow.

The only line of each testcase contains two integers n and s ($1 \leq n \leq s \leq 10^{18}$) — the number of days and the number of problems Polycarp wants to solve.

It's guaranteed that the distribution always exists within the given constraints.

Output

For each testcase print a single integer — the maximum value of a_n.

Example

input

```
3
1 15
3 9
2 6
```

output

```
15
4
4
```

Note

In the first testcase there is only one distribution: $[15]$.

In the second testcase the distribution that maximizes a_n is: $[2, 3, 4]$.

In the third testcase the distribution that maximizes a_n is: $[2, 4]$. $[3, 3]$ is a valid distribution but $a_n = 3$ which is smaller than 4. $[1, 5]$ is not a valid distribution because $5 > 2 \cdot 1$.

E. Palindromic Doubles

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

A subsequence is a sequence that can be obtained from another sequence by removing some elements without changing the order of the remaining elements.

A palindromic sequence is a sequence that is equal to the reverse of itself.

You are given a sequence of n integers $a_1, a_2, \ldots, a_n$. **Any integer value appears in a no more than twice**.

What is the length of the longest palindromic subsequence of sequence a?

Input
The first line contains a single integer t ($1 \le t \le 1000$) — the number of testcases.

Then the descriptions of t testcases follow.

The first line of each testcase contains a single integer n ($1 \le n \le 250\,000$) — the number of elements in the sequence.

The second line of each testcase contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$).

Any integer value appears in a no more than twice. The sum of n over all testcases doesn't exceed $250\,000$.

Output
For each testcase print a single integer — the length of the longest palindromic subsequence of sequence a.

Example

input
5 6 2 1 3 1 5 2 6 1 3 3 4 4 1 1 1 2 1 1 7 4 4 2 5 7 2 3
output
5 4 1 2 3

Note
Here are the longest palindromic subsequences for the example testcases:

- **2 1 3 1** 5 **2**
- **1 3 3** 4 4 **1** or **1** 3 3 **4 4 1**
- **1**
- **1 1**
- 4 4 **2 5** 7 **2** 3 or 4 4 **2** 5 **7 2** 3

F. Dogecoin

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

Recently, cryptocurrencies have become increasingly popular. Ann decided that it was about time she started mining the Dogecoin cryptocurrency. Ann's computer is not very powerful, so Ann mines exactly 1 Dogecoin per day. On the Internet, there are forecasts of the cost of 1 dogecoin for the next n days. Every day, Ann can sell any amount of dogecoin that she currently has. Note that if Ann mined Dogecoin on the i-th day, then she can sell it on the same day.

Ann has not yet decided when to start mining. She has prepared q possible plans and for each of them wants to know the maximum profit that she can get. Each of the plans is described by two numbers l and r — the first and last days of mining. Note that Ann should not have any Dogecoin left after the r-th day.

Input
The first line contains one integer n ($1 \le n \le 2 \cdot 10^5$).

The second line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le 10^6$), where c_i is the cost of Dogecoin on the i-th day.

The third line contains one integer q ($1 \le q \le 2 \cdot 10^5$) — the number of Ann's plans.

The following q lines contains two integers l and r ($1 \le l \le r \le n$) — the first and last days of mining.

Output
For each Ann's plan print one integer — the maximum profit that she can get using this plan.

Example

input
5

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

output

```
15 9 8 2
```

G. Painting Numbers

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

You are given n integers, each integer is from 1 to n, all of them are pairwise distinct. You have to paint them red and blue (each integer should have exactly one color).

The cost of painting is the number of pairs (x, y) such that $y \bmod x = 0$, y is red and x is blue.

For each $k \in [1, n]$, calculate the maximum cost of painting if exactly k integers should have a red color.

Input

The first line contains one integer n ($2 \le n \le 10^5$).

Output

For each $k \in [1, n]$ print one integer — the maximum cost of painting, if exactly k integers should be red.

Examples

input

```
6
```

output

```
3 5 6 6 5 0
```

input

```
2
```

output

```
1 0
```

input

```
11
```

output

```
3 6 9 11 12 13 14 14 13 10 0
```

H. Build From Suffixes

time limit per test: 10 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are given an integer n and a sequence a of $n - 1$ integers, each element is either 0 or 1.

You are asked to build a string of length n such that:

- each letter is one of "abcd";
- if $a_i = 1$, then the i-th suffix of the string is lexicographically smaller than the $(i + 1)$-th suffix;
- if $a_i = 0$, then the i-th suffix of the string is lexicographically greater than the $(i + 1)$-th suffix.

You will be asked q queries of form:

- i ($1 \le i \le n - 1$) — flip the value of a_i (if $a_i = 0$, then set a_i to 1, and vice versa).

After each query print the number of different strings that satisfy the given constraints modulo $998\,244\,353$.

Input

The first line contains two integers n and q ($2 \le n \le 10^5$; $1 \le q \le 10^5$) — the length of the string and the number of queries.

The second line contains $n - 1$ integers $a_1, a_2, \ldots, a_{n-1}$ ($a_i \in \{0, 1\}$) — the constraints on suffixes of the string.

Each of the next q lines contains a query: an integer i ($1 \le i \le n - 1$) — flip the value of a_i (if $a_i = 0$, then set a_i to 1, and vice

versa).

Output

After each query print the number of different strings that satisfy the given constraints modulo $998\,244\,353$.

Examples

input
2 2 0 1 1
output
6 10

input
3 4 0 0 1 2 1 2
output
20 10 14 20

input
10 10 0 0 0 1 1 1 0 1 0 4 1 8 4 2 9 5 6 6 8
output
1815 3201 2710 2776 2290 1644 2668 1271 2668 2436

Note

The i-th suffix of a string is a continuous substring that starts from the i-th position and ends in the last position.

A string a is lexicographically smaller than a string b if and only if one of the following holds:

- a is a prefix of b, but $a \neq b$;
- in the first position where a and b differ, the string a has a letter that appears earlier in the alphabet than the corresponding letter in b.

Two strings a and b of length n differ if there exists a position i such that $a_i \neq b_i$.

I. Demonic Invasion

time limit per test: 8 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Dalaran is a flying city of magic. It consists of n floating islands connected by m bridges (each bridge can be traversed in both directions). Every island is reachable from every other along these bridges. It would be a shame if some magic experiment went horribly wrong and this magnificent city got destroyed, right?

Well, just guess what happened. A portal to demons' realm was opened at the island 1. This is the start of day 1 of the catastrophe, and all k mages have gathered at the island 1 (they were trying to close the portal, but found out that it was impossible). At the start of day 2, a gigantic horde of demons will emerge from the portal, capture the island 1 and kill every mage staying at that

island. During each day i, if the island v was captured by demons by the start of the day, the horde of demons will travel across all bridges connected **directly to** v, capture every island they reach, and kill every mage they meet along the way.

Each bridge contains exactly one magic stone. Each mage at the start of the day can do one of the following actions:

- Spend one day travelling along one of the bridges connected to the island where the mage currently is. When passing through a bridge, the mage can pick up the magic stone from it (if there is any; each magic stone can be picked up by at most one mage). If the bridge is passed by the demons during the same day, or by the start of the next day, the island where the mage goes is already captured by the demonic horde (even if they arrive there at the same moment), the mage is killed.
- Perform a teleportation ritual to get to safety outside Dalaran. This ritual consumes two magic stones (and cannot be performed if the mage has less than two stones).

Each magic stone decays in 2 days, so if is picked up in the middle of day i, it decays in the middle of day $i + 2$. Decayed stones cannot be used in teleportation ritual.

Calculate the maximum number of mages that can get to safety.

Input
The first line contains three integers n, m and k ($2 \le n \le 10^5$; $n - 1 \le m \le 10^5$; $1 \le k \le 10^5$) — the number of islands, the number of bridges and the number of mages.

Then m lines follow, the i-th line contains two integers x_i and y_i ($1 \le x_i, y_i \le n$; $x_i \ne y_i$) denoting a bridge between the islands x_i and y_i.

Each pair of islands has at most one bridge connecting them. Every island is reachable from every other island along the bridges.

Output
Print one integer — the maximum number of mages that can get to safety.

Examples

input
4 4 1 1 2 2 3 3 4 4 1
output
1

input
4 4 4 1 2 2 3 3 4 4 1
output
2

input
3 2 10 1 2 2 3
output
1

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

J. Flower Shop

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Your friend is running a flower shop. In order to be prepared for the next holidays (when, as usual, sales skyrocket) she asked you to

write her a special program that will help to analyze the stocks she has.

There are n different types of flowers she can order and each flower of the type i costs w_i. The last holidays were a great success, she sold all flowers she had, so right now all her stocks are empty.

From this point, she starts routine operations of ordering and selling flowers, while trying to analyze what she has at hand. All of this can be represented as m queries of three types:

- "$1\ \ i\ \ c$" — she bought c flowers of type i;
- "$2\ \ i\ \ c$" — she disposed of c flowers of type i;
- "$3\ \ l\ \ r\ \ k$" — how many variants of bouquets she can make using only flowers of types $l, l+1, \ldots, r$ with the total cost no more than k. For simplicity, you can think that a bouquet is a multiset of flowers, and two bouquets are different if they are different as multisets. The cost of a bouquet is the sum of all flowers it has.

Help your friend and write the program that can process all these queries.

Input

The first line contains two integers n and m ($1 \le n \le 1000$; $1 \le m \le 1000$) — the number of flower types and the number of queries.

The second line contains n integers $w_1, w_2, \ldots, w_n$ ($1 \le w_i \le 1000$) — the cost of one flower of each type.

The next m lines contains queries — one per line. Each query has one of three types:

- $1\ \ i\ \ c$ ($1 \le i \le n$; $1 \le c \le 5000$);
- $2\ \ i\ \ c$ ($1 \le i \le n$; $1 \le c \le 5000$). It's guaranteed that there are at least c flowers of type i at this moment;
- $3\ \ l\ \ r\ \ k$ ($1 \le l \le r \le n$; $1 \le k \le 5000$)

It's guaranteed that the total cost of all flowers in stock after each query doesn't exceed 5000.

Output

For each query of the third type, print how many variants of bouquets she can make using only flowers of types $l, l+1, \ldots, r$ with the total cost no more than k. Since the answer may be too large, print it modulo $998\,244\,353$.

Example

input
5 12 1 2 3 2 1 1 1 5 1 2 3 1 3 1 3 1 5 10 3 4 5 100 1 4 4 1 5 1 3 2 5 7 3 1 1 3 3 1 5 100 2 1 5 3 1 5 100
output
40 0 28 3 479 79