Educational Codeforces Round 56 (Rated for Div. 2)

A. Dice Rolling

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

Mishka got a six-faced dice. It has integer numbers from 2 to 7 written on its faces (all numbers on faces are different, so this is an **almost** usual dice).

Mishka wants to get exactly x points by rolling his dice. The number of points is just a sum of numbers written at the topmost face of the dice for all the rolls Mishka makes.

Mishka doesn't really care about the number of rolls, so he just wants to know **any** number of rolls he can make to be able to get exactly x points for them. **Mishka is very lucky, so if the probability to get x points with chosen number of rolls is non-zero, he will be able to roll the dice in such a way.** Your task is to print this number. It is **guaranteed** that at least one answer exists.

Mishka is also very curious about different number of points to score so you have to answer t **independent** queries.

Input

The first line of the input contains one integer t ($1 \le t \le 100$) — the number of queries.

Each of the next t lines contains one integer each. The i-th line contains one integer x_i ($2 \le x_i \le 100$) — the number of points Mishka wants to get.

Output

Print t lines. In the i-th line print the answer to the i-th query (i.e. **any** number of rolls Mishka can make to be able to get exactly x_i points for them). It is **guaranteed** that at least one answer exists.

Example

input

```
4
2
13
37
100
```

output

```
1
3
8
27
```

Note

In the first query Mishka can roll a dice once and get 2 points.

In the second query Mishka can roll a dice 3 times and get points 5, 5 and 3 (for example).

In the third query Mishka can roll a dice 8 times and get 5 points 7 times and 2 points with the remaining roll.

In the fourth query Mishka can roll a dice 27 times and get 2 points 11 times, 3 points 6 times and 6 points 10 times.

B. Letters Rearranging

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

You are given a string s consisting only of lowercase Latin letters.

You can rearrange all letters of this string as you wish. Your task is to obtain a **good** string by rearranging the letters of the given string or report that it is impossible to do it.

Let's call a string **good** if it is not a palindrome. Palindrome is a string which is read from left to right the same as from right to left. For example, strings "abacaba", "aa" and "z" are palindromes and strings "bba", "xd" are not.

You have to answer t **independent** queries.

Input

The first line of the input contains one integer t ($1 \le t \le 100$) — number of queries.

Each of the next t lines contains one string. The i-th line contains a string s_i consisting only of lowercase Latin letter. It is guaranteed that the **length** of s_i is **from** 1 **to** 1000 (inclusive).

Output

Print t lines. In the i-th line print the answer to the i-th query: `-1` if it is impossible to obtain a **good** string by rearranging the letters of s_i and **any good** string which can be obtained from the given one (by rearranging the letters) otherwise.

Example

input

```
3
aa
abacaba
xdd
```

output

```
-1
abaacba
xdd
```

Note

In the first query we cannot rearrange letters to obtain a **good** string.

Other examples (not all) of correct answers to the second query: "ababaca", "abcabaa", "baacaba".

In the third query we can do nothing to obtain a **good** string.

C. Mishka and the Last Exam

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

Mishka is trying really hard to avoid being kicked out of the university. In particular, he was doing absolutely nothing for the whole semester, miraculously passed some exams so that just one is left.

There were n classes of that subject during the semester and on i-th class professor mentioned some non-negative integer a_i to the students. It turned out, the exam was to tell the whole sequence back to the professor.

Sounds easy enough for those who attended every class, doesn't it?

Obviously Mishka didn't attend any classes. However, professor left some clues on the values of a to help out students like Mishka:

- a was sorted in non-decreasing order ($a_1 \le a_2 \le \dots \le a_n$);
- n was even;
- the following sequence b, consisting of $\frac{n}{2}$ elements, was formed and given out to students: $b_i = a_i + a_{n-i+1}$.

Professor also mentioned that any sequence a, which produces sequence b with the presented technique, will be acceptable.

Help Mishka to pass that last exam. Restore any sorted sequence a of non-negative integers, which produces sequence b with the presented technique. It is guaranteed that there exists at least one correct sequence a, which produces the given sequence b.

Input

The first line contains a single integer n ($2 \le n \le 2 \cdot 10^5$) — the length of sequence a. n is always even.

The second line contains $\frac{n}{2}$ integers $b_1, b_2, \dots, b_{\frac{n}{2}}$ ($0 \le b_i \le 10^{18}$) — sequence b, where $b_i = a_i + a_{n-i+1}$.

It is guaranteed that there exists at least one correct sequence a, which produces the given sequence b.

Output

Print n integers $a_1, a_2, \dots, a_n$ ($0 \le a_i \le 10^{18}$) in a single line.

$a_1 \le a_2 \le \dots \le a_n$ should be satisfied.

$b_i = a_i + a_{n-i+1}$ should be satisfied for all valid i.

Examples

input

```
4
5 6
```

output

```
2 3 3 3
```

input

```
6
2 1 2
```

output

```
0 0 1 1 1 2
```

D. Beautiful Graph

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

You are given an undirected unweighted graph consisting of n vertices and m edges.

You have to write a number on each vertex of the graph. Each number should be 1, 2 or 3. The graph becomes beautiful if for each edge the sum of numbers on vertices connected by this edge is odd.

Calculate the number of possible ways to write numbers 1, 2 and 3 on vertices so the graph becomes beautiful. Since this number may be large, print it modulo 998244353.

Note that you have to write exactly one number on each vertex.

The graph does not have any self-loops or multiple edges.

Input

The first line contains one integer t ($1 \le t \le 3 \cdot 10^5$) — the number of tests in the input.

The first line of each test contains two integers n and m ($1 \le n \le 3 \cdot 10^5, 0 \le m \le 3 \cdot 10^5$) — the number of vertices and the number of edges, respectively. Next m lines describe edges: i-th line contains two integers u_i, v_i ($1 \le u_i, v_i \le n; u_i \ne v_i$) — indices of vertices connected by i-th edge.

It is guaranteed that $\sum\limits_{i=1}^{t} n \le 3 \cdot 10^5$ and $\sum\limits_{i=1}^{t} m \le 3 \cdot 10^5$.

Output

For each test print one line, containing one integer — the number of possible ways to write numbers 1, 2, 3 on the vertices of given graph so it becomes beautiful. Since answers may be large, print them modulo 998244353.

Example

input

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

output

```
4
0
```

Note

Possible ways to distribute numbers in the first test:

1. the vertex 1 should contain 1, and 2 should contain 2;
2. the vertex 1 should contain 3, and 2 should contain 2;
3. the vertex 1 should contain 2, and 2 should contain 1;
4. the vertex 1 should contain 2, and 2 should contain 3.

In the second test there is no way to distribute numbers.

E. Intersection of Permutations

time limit per test: 6 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are given two permutations a and b, both consisting of n elements. Permutation of n elements is such a integer sequence that each value from 1 to n appears exactly once in it.

You are asked to perform two types of queries with them:

- $1\ l_a\ r_a\ l_b\ r_b$ — calculate the number of values which appear in both segment $[l_a; r_a]$ of positions in permutation a and segment $[l_b; r_b]$ of positions in permutation b;
- $2\ x\ y$ — swap values on positions x and y in permutation b.

Print the answer for each query of the first type.

It is guaranteed that there will be at least one query of the first type in the input.

Input

The first line contains two integers n and m ($2 \le n \le 2 \cdot 10^5$, $1 \le m \le 2 \cdot 10^5$) — the number of elements in both permutations and the number of queries.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le n$) — permutation a. It is guaranteed that each value from 1 to n appears in a exactly once.

The third line contains n integers $b_1, b_2, \dots, b_n$ ($1 \le b_i \le n$) — permutation b. It is guaranteed that each value from 1 to n appears in b exactly once.

Each of the next m lines contains the description of a certain query. These are either:

- $1\ l_a\ r_a\ l_b\ r_b$ ($1 \le l_a \le r_a \le n$, $1 \le l_b \le r_b \le n$);
- $2\ x\ y$ ($1 \le x, y \le n$, $x \ne y$).

Output

Print the answers for the queries of the first type, each answer in the new line — the number of values which appear in both segment $[l_a; r_a]$ of positions in permutation a and segment $[l_b; r_b]$ of positions in permutation b.

Example

input

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

output

```
1
1
1
2
0
```

Note

Consider the first query of the first example. Values on positions $[1; 2]$ of a are $[5, 1]$ and values on positions $[4; 5]$ of b are $[1, 4]$. Only value 1 appears in both segments.

After the first swap (the second query) permutation b becomes $[2, 1, 3, 5, 4, 6]$.

After the second swap (the sixth query) permutation b becomes $[5, 1, 3, 2, 4, 6]$.

F. Vasya and Array

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Vasya has got an array consisting of n integers, and two integers k and len in addition. All numbers in the array are either between 1 and k (inclusive), or equal to -1. The array is good if there is no segment of len consecutive **equal** numbers.

Vasya will replace each -1 with some number from 1 to k (inclusive) in such a way that the resulting array is good. Tell him the number of ways to do this replacement. Since the answer may be large, print it modulo 998244353.

Input

The first line contains three integers n, k and len ($1 \le n \le 10^5, 1 \le k \le 100, 1 \le len \le n$).

The second line contains n numbers — the array. Each number is either -1 or between 1 and k (inclusive).

Output

Print one integer — the number of ways to replace each -1 with some number from 1 to k (inclusive) so the array is good. The answer may be large, so print it modulo 998244353.

Examples

input

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

output

```
2
```

input

```
6 3 2
1 1 -1 -1 -1 -1
```

output

```
0
```

input

```
10 42 7
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1
```

output

```
645711643
```

Note

Possible answers in the first test:

1. $[1, 2, 1, 1, 2]$;
2. $[1, 2, 1, 2, 2]$.

There is no way to make the array good in the second test, since first two elements are equal.

There are too many answers in the third test, so we won't describe any of them.

G. Multidimensional Queries

time limit per test: 6 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are given an array a of n points in k-dimensional space. Let the distance between two points a_x and a_y be $\sum\limits_{i=1}^{k} |a_{x,i} - a_{y,i}|$ (it is also known as Manhattan distance).

You have to process q queries of the following two types:

- 1 i b_1 b_2 ... b_k — set i-th element of a to the point $(b_1, b_2, \dots, b_k)$;
- 2 l r — find the maximum distance between two points a_i and a_j, where $l \le i, j \le r$.

Input

The first line contains two numbers n and k ($1 \le n \le 2 \cdot 10^5$, $1 \le k \le 5$) — the number of elements in a and the number of dimensions of the space, respectively.

Then n lines follow, each containing k integers $a_{i,1}$, $a_{i,2}$, ..., $a_{i,k}$ ($-10^6 \le a_{i,j} \le 10^6$) — the coordinates of i-th point.

The next line contains one integer q ($1 \le q \le 2 \cdot 10^5$) — the number of queries.

Then q lines follow, each denoting a query. There are two types of queries:

- 1 i b_1 b_2 ... b_k ($1 \le i \le n$, $-10^6 \le b_j \le 10^6$) — set i-th element of a to the point $(b_1, b_2, \dots, b_k)$;
- 2 l r ($1 \le l \le r \le n$) — find the maximum distance between two points a_i and a_j, where $l \le i, j \le r$.

There is at least one query of the second type.

Output

Print the answer for each query of the second type.

Example

input

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

```
2 1 5
1 4 -1 -2
2 1 5
```

output

```
8
4
4
12
10
```