Codeforces Round 993 (Div. 4)

A. Easy Problem

1 second, 256 megabytes

Cube is given an integer n. She wants to know how many ordered pairs of positive integers (a, b) there are such that $a = n - b$. Since Cube is not very good at math, please help her!

Input

The first line contains an integer t ($1 \le t \le 99$) — the number of test cases.

The only line of each test case contains an integer n ($2 \le n \le 100$).

Output

For each test case, output the number of ordered pairs (a, b) on a new line.

input
3 2 4 6
output
1 3 5

In the first test case, the only ordered pair that works is $(a, b) = (1, 1)$.

In the second test case, the three ordered pairs of (a, b) that work are $(3, 1), (2, 2), (1, 3)$.

B. Normal Problem

1 second, 256 megabytes

A string consisting of only characters 'p', 'q', and 'w' is painted on a glass window of a store. Ship walks past the store, standing directly in front of the glass window, and observes string a. Ship then heads inside the store, looks directly at the same glass window, and observes string b.

Ship gives you string a. Your job is to find and output b.

Input

The first line contains an integer t ($1 \le t \le 100$) — the number of test cases.

The only line of each test case contains a string a ($1 \le |a| \le 100$) — the string Ship observes from outside the store. It is guaranteed that a only contains characters 'p', 'q', and 'w'.

Output

For each test case, output string b, the string Ship observes from inside the store, on a new line.

input
5 qwq ppppp pppwwwqqq wqpqwpqwwqp pqpqpqpq
output
pwp qqqqq pppwwwqqq qpwwpqwpqpw pqpqpqpq

C. Hard Problem

1 second, 256 megabytes

Ball is the teacher in Paperfold University. The seats of his classroom are arranged in 2 rows with m seats each.

Ball is teaching $a + b + c$ monkeys, and he wants to assign as many monkeys to a seat as possible. Ball knows that a of them only want to sit in row 1, b of them only want to sit in row 2, and c of them have no preference. Only one monkey may sit in each seat, and each monkey's preference must be followed if it is seated.

What is the maximum number of monkeys that Ball can seat?

Input

The first line contains an integer t ($1 \le t \le 10^4$) — the number of test cases.

Each test case contains four integers m, a, b, and c ($1 \le m, a, b, c \le 10^8$).

Output

For each test case, output the maximum number of monkeys you can seat.

input
5 10 5 5 10 3 6 1 1 15 14 12 4 1 1 1 1 420 6 9 69
output
20 5 30 2 84

In the second test case, 6 monkeys want to sit in the front row, but only 3 seats are available. The monkeys that have no preference and the monkeys who prefer sitting in the second row can sit in the second row together. Thus, the answer is $3 + 2 = 5$.

D. Harder Problem

2 seconds, 256 megabytes

Given a sequence of positive integers, a positive integer is called a *mode* of the sequence if it occurs the maximum number of times that any positive integer occurs. For example, the mode of $[2, 2, 3]$ is 2. Any of 9, 8, or 7 can be considered to be a mode of the sequence $[9, 9, 8, 8, 7, 7]$.

You gave UFO an array a of length n. To thank you, UFO decides to construct another array b of length n such that a_i is a mode of the sequence $[b_1, b_2, \ldots, b_i]$ for all $1 \le i \le n$.

However, UFO doesn't know how to construct array b, so you must help her. Note that $1 \le b_i \le n$ must hold for your array for all $1 \le i \le n$.

Input

The first line contains t ($1 \le t \le 10^4$) — the number of test cases.

The first line of each test case contains an integer n ($1 \le n \le 2 \cdot 10^5$) — the length of a.

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

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 n numbers $b_1, b_2, \ldots, b_n$ ($1 \le b_i \le n$) on a new line. It can be shown that b can always be constructed. If there are multiple possible arrays, you may print any.

input

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

output

```
1 2
1 1 2 2
4 5 5 1 1 2 2 3
1 8 2 2 1 3 3 9 1 1
```

Let's verify the correctness for our sample output in test case 2.

- At $i = 1$, 1 is the only possible mode of [1].
- At $i = 2$, 1 is the only possible mode of [1, 1].
- At $i = 3$, 1 is the only possible mode of [1, 1, 2].
- At $i = 4$, 1 or 2 are both modes of [1, 1, 2, 2]. Since $a_i = 2$, this array is valid.

E. Insane Problem

2 seconds, 256 megabytes

Wave is given five integers k, l_1, r_1, l_2, and r_2. Wave wants you to help her count the number of ordered pairs (x, y) such that all of the following are satisfied:

- $l_1 \le x \le r_1$.
- $l_2 \le y \le r_2$.
- There exists a non-negative integer n such that $\frac{y}{x} = k^n$.

Input

The first line contains an integer t ($1 \le t \le 10^4$) — the number of test cases.

The only line of each test case contains five integers k, l_1, r_1, l_2, and r_2 ($2 \le k \le 10^9$, $1 \le l_1 \le r_1 \le 10^9$, $1 \le l_2 \le r_2 \le 10^9$).

Output

For each test case, output the number of matching ordered pairs (x, y) on a new line.

input

```
5
2 2 6 2 12
2 1 1000000000 1 1000000000
3 5 7 15 63
1000000000 1 5 6 1000000000
15 17 78 2596 20914861
```

output

```
12
1999999987
6
1
197
```

In the third test case, the matching ordered pairs are the following:

- (5, 15)
- (5, 45)
- (6, 18)
- (6, 54)
- (7, 21)
- (7, 63)

In the fourth test case, the only valid ordered pair is (1, 1 000 000 000)

F. Easy Demon Problem

4 seconds, 256 megabytes

For an arbitrary grid, Robot defines its *beauty* to be the sum of elements in the grid.

Robot gives you an array a of length n and an array b of length m. You construct a n by m grid M such that $M_{i,j} = a_i \cdot b_j$ for all $1 \le i \le n$ and $1 \le j \le m$.

Then, Robot gives you q queries, each consisting of a single integer x. For each query, determine whether or not it is possible to perform the following operation **exactly** once so that M has a *beauty* of x:

1. Choose integers r and c such that $1 \le r \le n$ and $1 \le c \le m$
2. Set $M_{i,j}$ to be 0 for all ordered pairs (i, j) such that $i = r$, $j = c$, or both.

Note that queries are **not persistent**, meaning that you do not actually set any elements to 0 in the process — you are only required to output if it is possible to find r and c such that if the above operation is performed, the *beauty* of the grid will be x. Also, note that you must perform the operation for each query, even if the beauty of the original grid is already x.

Input

The first line contains three integers n, m, and q ($1 \le n, m \le 2 \cdot 10^5$, $1 \le q \le 5 \cdot 10^4$) — the length of a, the length of b, and the number of queries respectively.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le |a_i| \le n$).

The third line contains m integers $b_1, b_2, \ldots, b_m$ ($0 \le |b_i| \le m$).

The following q lines each contain a single integer x ($1 \le |x| \le 2 \cdot 10^5$), the beauty of the grid you wish to achieve by setting all elements in a row and a column to 0.

Output

For each testcase, output "YES" (without quotes) if there is a way to perform the aforementioned operation such that the beauty is x, and "NO" (without quotes) otherwise.

You can output "YES" and "NO" in any case (for example, strings "yES", "yes" and "Yes" will be recognized as a positive response).

input

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

output

```
NO
YES
NO
NO
YES
NO
```

input

```
5 5 6
1 -2 3 0 0
0 -2 5 0 -3
4
-3
5
2
-1
2
```

output

```
YES
YES
YES
YES
NO
YES
```

In the second example, the grid is

```
0 -2 5 0 -3

0 4 -10 0 6

0 -6 15 0 -9

0 0 0 0 0

0 0 0 0 0
```

By performing the operation with $r = 4$ and $c = 2$, we create the following grid:

```
0 0 5 0 -3

0 0 -10 0 6

0 0 15 0 -9

0 0 0 0 0

0 0 0 0 0
```

which has *beauty* 4. Thus, we output `YES`.

In the second query, selecting $r = 3$ and $c = 5$ creates a grid with *beauty* -3.

In the third query, selecting $r = 3$ and $c = 3$ creates a grid with *beauty* 5.

G1. Medium Demon Problem (easy version)

2 seconds, 256 megabytes

This is the easy version of the problem. The key difference between the two versions is highlighted in bold.

A group of n spiders has come together to exchange plushies. Initially, each spider has 1 plushie. Every year, if spider i has at least one plushie, he will give exactly one plushie to spider r_i. Otherwise, he will do nothing. Note that all plushie transfers happen at the same time. **In this version, if any spider has more than 1 plushie at any point in time, they will throw all but 1 away.**

The process is *stable* in the current year if each spider has the same number of plushies (before the current year's exchange) as he did the previous year (before the previous year's exchange). Note that year 1 can never be *stable*.

Find the first year in which the process becomes *stable*.

Input

The first line contains an integer t ($1 \le t \le 10^4$) — the number of test cases.

The first line of each test case contains an integer n ($2 \le n \le 2 \cdot 10^5$) — the number of spiders.

The following line contains n integers $r_1, r_2, \ldots, r_n$ ($1 \le r_i \le n, r_i \ne i$) — the recipient of the plushie of each spider.

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 an integer on a new line, the first year in which the process becomes *stable*.

input
5 2 2 1 5 2 3 4 5 1 5 2 1 4 2 3 5 4 1 1 5 4 10 4 3 9 1 6 7 9 10 10 3

output
2 2 5 4 5

For the second test case:

- At year 1, the following array shows the number of plushies each spider has: [1, 1, 1, 1, 1]. Then, year 1's exchange happens.
- At year 2, the following array shows the number of plushies each spider has: [1, 1, 1, 1, 1]. Since this array is the same as the previous year, this year is *stable*.

For the third test case:

- At year 1, the following array shows the number of plushies each spider has: [1, 1, 1, 1, 1]. Then, year 1's exchange happens.
- At year 2, the following array shows the number of plushies each spider has: [1, 1, 1, 1, 0]. Then, year 2's exchange happens. Note that even though two spiders gave spider 2 plushies, spider 2 may only keep one plushie.
- At year 3, the following array shows the number of plushies each spider has: [1, 1, 0, 1, 0]. Then, year 3's exchange happens.
- At year 4, the following array shows the number of plushies each spider has: [1, 1, 0, 0, 0]. Then, year 4's exchange happens.
- At year 5, the following array shows the number of plushies each spider has: [1, 1, 0, 0, 0]. Since this array is the same as the previous year, this year is *stable*.

G2. Medium Demon Problem (hard version)

2 seconds, 256 megabytes

This is the hard version of the problem. The key difference between the two versions is highlighted in bold.

A group of n spiders has come together to exchange plushies. Initially, each spider has 1 plushie. Every year, if spider i has at least one plushie, he will give exactly one plushie to spider r_i. Otherwise, he will do nothing. Note that all plushie transfers happen at the same time. **In this version, each spider is allowed to have more than 1 plushie at any point in time.**

The process is *stable* in the current year if each spider has the same number of plushies (before the current year's exchange) as he did the previous year (before the previous year's exchange). Note that year 1 can never be *stable*.

Find the first year in which the process becomes *stable*.

Input

The first line contains an integer t ($1 \le t \le 10^4$) — the number of test cases.

The first line of each test case contains an integer n ($2 \le n \le 2 \cdot 10^5$) — the number of spiders.

The following line contains n integers $r_1, r_2, \ldots, r_n$ ($1 \le r_i \le n, r_i \ne i$) — the recipient of the plushie of each spider.

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 an integer on a new line, the first year in which the process becomes *stable*.

input

```
5
2
2 1
5
2 3 4 5 1
5
2 1 4 2 3
5
4 1 1 5 4
10
4 3 9 1 6 7 9 10 10 3
```

output

```
2
2
5
5
5
```

For the second test case:

- At year 1, the following array shows the number of plushies each spider has: [1, 1, 1, 1, 1]. Then, year 1's exchange happens.
- At year 2, the following array shows the number of plushies each spider has: [1, 1, 1, 1, 1]. Since this array is the same as the previous year, this year is *stable*.

For the third test case:

- At year 1, the following array shows the number of plushies each spider has: [1, 1, 1, 1, 1]. Then, year 1's exchange happens.
- At year 2, the following array shows the number of plushies each spider has: [1, 2, 1, 1, 0]. Then, year 2's exchange happens.
- At year 3, the following array shows the number of plushies each spider has: [1, 3, 0, 1, 0]. Then, year 3's exchange happens.
- At year 4, the following array shows the number of plushies each spider has: [1, 4, 0, 0, 0]. Then, year 4's exchange happens.
- At year 5, the following array shows the number of plushies each spider has: [1, 4, 0, 0, 0]. Since this array is the same as the previous year, this year is *stable*.

H. Hard Demon Problem

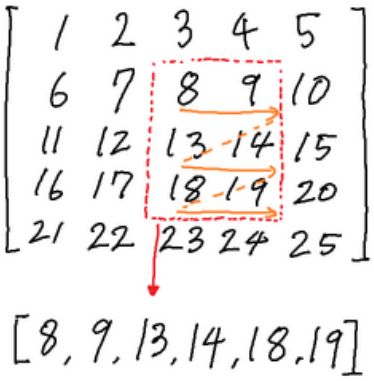
3.5 seconds, 512 megabytes

Swing is opening a pancake factory! A good pancake factory must be good at flattening things, so Swing is going to test his new equipment on 2D matrices.

Swing is given an $n \times n$ matrix M containing positive integers. He has q queries to ask you.

For each query, he gives you four integers x_1, y_1, x_2, y_2 and asks you to flatten the submatrix bounded by (x_1, y_1) and (x_2, y_2) into an array A. Formally,
$A = [M_{(x1,y1)}, M_{(x1,y1+1)}, \ldots, M_{(x1,y2)}, M_{(x1+1,y1)}, M_{(x1+1,y1+1)}, \ldots, M_{(x2,y2)}]$.

The following image depicts the flattening of a submatrix bounded by the red dotted lines. The orange arrows denote the direction that the elements of the submatrix are appended to the back of A, and A is shown at the bottom of the image.



Afterwards, he asks you for the value of $\sum_{i=1}^{|A|} A_i \cdot i$ (sum of $A_i \cdot i$ over all i).

Input

The first line contains an integer t ($1 \le t \le 10^3$) — the number of test cases.

The first line of each test contains two integers n and q ($1 \le n \le 2000$, $1 \le q \le 10^6$) — the length of M and the number of queries.

The following n lines contain n integers each, the i'th of which contains $M_{(i,1)}, M_{(i,2)}, \ldots, M_{(i,n)}$ ($1 \le M_{(i,j)} \le 10^6$).

The following q lines contain four integers x_1, y_1, x_2, and y_2 ($1 \le x_1 \le x_2 \le n$, $1 \le y_1 \le y_2 \le n$) — the bounds of the query.

It is guaranteed that the sum of n over all test cases does not exceed 2000 and the sum of q over all test cases does not exceed 10^6.

Output

For each test case, output the results of the q queries on a new line.

input

```
2
4 3
1 5 2 4
4 9 5 3
4 5 2 3
1 5 5 2
1 1 4 4
2 2 3 3
1 2 4 3
3 3
1 2 3
4 5 6
7 8 9
1 1 1 3
1 3 3 3
2 2 2 2
```

output

```
500 42 168
14 42 5
```

In the second query of the first test case, $A = [9, 5, 5, 2]$. Therefore, the sum is $1 \cdot 9 + 2 \cdot 5 + 3 \cdot 5 + 4 \cdot 2 = 42$.