National Taiwan University

2014 NTU Final

A	Another Easy Problem	10 s
B	Balance	3 s
C	Cool lucky function	3 s
D	Dreamoon and his happy singer friends	5 s
E	Expensive Mop	10 s
F	Forbidden substring	3 s
G	Guava Tower	3 s
H	Hungry TWT	10 s
I	Instant Noodles	3 s
J	Just Composite	7 s

December 27, 2014

(almost empty page)

Another Easy Problem

Description

Being a adept ACM competitor, you have solved many easy problems such as **COT**, **COT2**, **COT3** and **COT4**. Here is another easy problem for you :) Consider a rooted tree with n nodes, where node 1 is the root of that tree, and each node has its own value a_i. You need to handle four types of queries:

- `1 u v` Ask the XOR sum and the maximum value over the path from u to v.
- `2 u` Ask the XOR sum and the maximum value over the subtree rooted at u.
- `3 u v w` Change all values of nodes to w over the path from u to v.
- `4 u w` Change all values of nodes to w over the subtree rooted at u.

Fcrh think it's quite easy. How about you?

Input

The first line contains an integer T indicating the total number of test cases. Each test case starts with one line containing an integer n. Then one line contains n integers $a_1, a_2, \ldots, a_n$. Then $n-1$ lines, each line contains two integers x, y, which means there is an edge between x and y. Then one line contains an integer q indicating the number of queries. Then q lines, each line contains one query described above.

- $1 \leq T \leq 1000$
- $1 \leq n, q \leq 10^5$
- $0 \leq a_i, w \leq 10^9$
- $1 \leq x, y, u, v \leq n$
- There are at most 5 test cases with $n + q > 2000$

Output

For each query of type 1 and 2, please output two integers indicating the XOR sum and the maximum value.

Sample Input

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

Sample Output

```
0 5
0 5
0 5
1 5
0 2
2 2
```

Balance

Description

As a "Balance" lover, you wear New Balance shoes, and code a balanced binary search tree everyday.

One day you pick up an undirected graph G on the way to National Taiwan University (don't ask me how), and you want to assign a direction to each edge (don't ask me why). As a "Balance" lover, you also want the modified graph is balanced. That is, the difference between in-degree and out-degree is no more than 1 for each node. In other words, the modified graph must satisfy $|d_i^+ - d_i^-| \leq 1$ for each node i. How to do that?

Input

The first line contains an integer T indicating the total number of test cases. Each test case starts with one line containing two integers n, m, denoting the number of nodes and the number of edges in the undirected graph G. Then m lines, each contains 2 integers a_i, b_i, denoting an edge (a_i, b_i) in G.

- $1 \leq T \leq 1000$
- $2 \leq n \leq 10^5$
- $1 \leq m \leq 2 \times 10^5$
- $0 \leq a_i, b_i < n$
- $a_i \neq b_i$, but (a_i, b_i) can be equal to (a_j, b_j)
- There are at most 5 test cases with $n + m > 2000$

Output

For each test case, please output one line with m characters which shows a balancing assignment. If the i-th character is `0`, the direction of that edge is $a_i \to b_i$; Otherwise the i-th character must be `1`, which indicates the direction is $b_i \to a_i$. If there is no valid assignment, please take a deep breath and rethink about this problem.

Sample Input

```
2
3 2
0 1
2 1
10 7
5 0
2 1
6 5
1 4
3 1
4 1
5 9
```

Sample Output

```
01
1110000
```

Cool lucky function

Description

Pipi is playing an interesting game. He has a lucky basis which contains n positive integers $a_1, a_2, \ldots, a_n$. Each morning he will randomly choose two numbers x, y from the lucky basis, and calculate how lucky is today for him by the following function:

```
int lucky(int x, int y) {
    if (x > y) return lucky(y, x);
    if (!x || !y) return x + y;
    if (x & 1) {
        if (y & 1) return lucky((y - x) >> 1, x);
        else return lucky(x, y >> 1);
    } else {
        if (y & 1) return lucky(x >> 1, y);
        else return lucky(x >> 1, y >> 1) << 1;
    }
}
```

If the result is 1, he will think today is incredibly unlucky. Can you determine whether it is possible that he get incredibly unlucky given the lucky basis?

Input

The first line contains an integer T indicating the total number of test cases. Each test case starts with one line containing an integer n. Then one line contains n integers $a_1, a_2, \ldots, a_n$.

- $1 \leq T \leq 20$
- $2 \leq n \leq 10^5$
- $1 \leq a_i \leq 10^6$

Output

For each test case, output `777` if he always feel lucky; otherwise, output `QQQ`.

Sample Input

```
2
3
2 3 4
4
6 10 15 30
```

Sample Output

```
QQQ
777
```

Dreamoon and his happy singer friends

Description

Dreamoon is a good man who likes singing very much. He has lots of friends who likes to sing, too. Now he wants to invite all of his friends to sing a beautiful song together. The song has a wide vocal range. For simplicity, we label pitches from lowest to highest with integers 1 through k.

The song consists of m consecutive notes, i.e. no two notes is sung at the same time. The pitch of the i-th note is a_i. Dreamoon invites $n-1$ friends and all of them has their own vocal range: the i-th person can sing pitches in the range $[l_i, r_i]$.

They want to sing in the following manner: all notes of the same pitch must be sung by exactly one specified person, and of course this pitch must fit into the singer's vocal range. Now Dreamoon wants to arrange the pitch assignment such that the occurrence of "two consecutive notes in the song are sung by two different persons" is minimized for the best quality of the music. Dreamoon is sure that there exists at least one valid assignment for the song because they are professional … sort of.

And then Dreamoon gives the assignment task to the most brilliant man in the group, Shik. But Shik thinks this task is way too easy to spend time on it, he's busy dealing with some more challenging tasks (for example, the difficult problems in the PK)! So Shik asks Drazil if he could kindly do the assignment. At last, Drazil needs your help! Please help Drazil on solving this task!

Input

The first line of the input contains one integer T, indicating the number of test cases. The first line of each test case contains three integers corresponding to n, m, k in the statement. The second line of each test case contains m integers $a_1, a_2, \ldots, a_m$. For the following n lines, i-th line contains two integers l_i and r_i.

- $1 \leq n, m, k \leq 200000$
- $1 \leq a_i \leq k$
- $1 \leq l_i \leq r_i \leq k$
- $\sum n, \sum m, \sum k \leq 2000000$ if we sum over all test cases.

Output

For each test case, output a single line containing one integer, indicating the minimum number of two consecutive notes singed by two different persons.

Sample Input

```
3
3 7 3
1 2 3 2 1 2 3
1 2
2 3
3 3
3 7 3
1 2 3 2 1 2 3
1 2
2 3
1 3
3 7 3
1 2 3 2 1 2 3
1 1
2 2
3 3
```

Sample Output

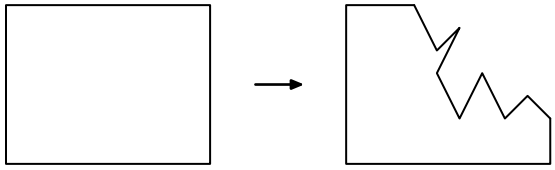
```
3
0
6
```

Expensive Mop

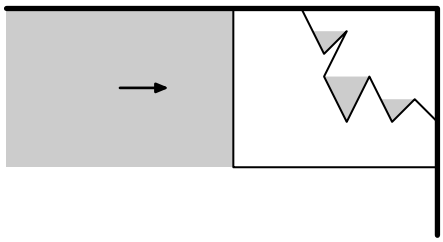
Description

Freshman tmt514 came to the hostel and he cannot endure how dirty is the room. There was a thick layer of dust on the floor, so he decided to buy a expensive mop to wash the floor.

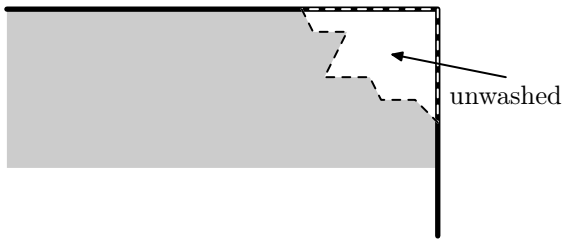
Originally the mop was a perfect rectangle, but when transporting from the store to the hostel he broke one of its corners. Thus the mop became a cool polygon.



He moped the floor from one side of the large rectangular room to another, without taking it away from the wall.



As you know, the mop was broken, so one corner of the room was left unwashed.



Please help him to calculate the area of the unwashed floor.

Input

The first line contains an integer T indicating the total number of test cases. Each test case starts with one line containing two integers w and h, denoting the size of mop. Then one line contains an integer n, denoting the number of vertices that connects the adjacent sides of the broken corner. Each of the following n lines contains two integers x_i and y_i, denoting the coordinates of the i-th vertices of the broken line.

Assume that the mop was touching the wall at line $y = h$.

- $1 \leq T \leq 60$
- $2 \leq w, h, n \leq 10^5$
- $y_1 = h$
- $x_n = w$
- $0 < x_i < w, \forall 1 \leq i < n$
- $0 < y_i < h, \forall 1 < i \leq n$
- The broken line does not intersect or touch itself.

Output

Please output the area of the unwashed floor. The answer will be considered correct if the absolute or relative error does not exceed 10^{-6}.

Sample Input

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

Sample Output

```
18.0
```

Forbidden substring

Description

Do you remember the "Beautiful substring"? Let's recall the problem.

> Fcrh is an expert of pronouncing string studying in the National Tomato University. In his world, each string is either beautiful or ugly. Whenever he is asked to pronounce a string, he will check whether the string is beautiful and reject to do the job if it's not. What make things worse is that any substring of a ugly string is also ugly! So, sometimes it's pretty hard to ask Fcrh to pronounce strings for you.
>
> To make things more efficient, Fcrh maintains a list of ugly strings. One can check whether a string is ugly with the list by himself, hence Fcrh won't be bother by those ugly strings.
>
> Shik, a string lover in the ICPC club, find an interesting ICPC task: Given a string S, how many kinds of substring of S is not ugly in Fcrh's opinion according to the list?

With the help of dreamoon, Shik solved the problem with an very efficient algorithm. However, solving only one problem is not enough for a string lover like Shik. Hence, Shik design a brand new ICPC task: Given a string S, how may kinds of substring of S is a substring of exactly one of the string in Fcrh's list.

Input

The first line contains an integer T indicating the total number of test cases. Each test case starts with a line containing the string S. The next line contains an integer N, denoting Fcrh has added N strings into the list. Each of the next N lines contains a string F_i, the i-th string Fcrh add into the list.

- $1 \leq T \leq 100$
- $1 \leq |S| \leq 2 \times 10^5$
- $1 \leq \sum |F_i| \leq 10^6$
- Strings are consist of English lowercase letters.

There're only 10% of test cases with $|S| > 2 \times 10^4$ or $\sum |F_i| > 10^5$.

Output

For each F_i, output an integer indicating how many kinds of substring of S is also a substring of exactly one string in Fcrh's list after adding F_i.

Sample Input

```
1
aaab
3
aaa
aab
aaab
```

Sample Output

```
3
4
1
```

Guava Tower

Description

Left the puzzle, or destroy the world

Guava kingdom is the strongest kingdom among those fruit kingdoms. Although guava kingdom is famous with it's technology and most of it's citizens are rational, there are still some rumors and legends. The most popular one is the legend of guava tower, which says that while the puzzle of guava tower is solved, god of guava, tmt, will come and destroy the world.

As the smartest scientist living in the guava kingdom, you'd like to disprove anything seems wrong to you. So, you decided to solve the guava tower puzzle.

The guava tower is a puzzle similar to the hanoi tower. There're three pegs and n disks with size from 1 to n on the first peg. However, unlike the hanoi tower, you don't need to move those disks physically. Instead, you need to find a way to answer whether a sequence of movement of disks is a solution to the corresponding hanoi tower.

The i-th number in the sequence is the size of disk that you should move at the i-th step. You may move the disk to any peg (but not the original one) as long as you obey the rule of hanoi tower. A sequence is a solution means that it's possible to solve the hanoi tower by moving disks in the way the sequence specify.

Since the computation power of computer may help you solve this problem, you decided to write a program that can answer the question.

Input

The first line contains an integer T indicating the total number of test cases. Each test case starts with a line containing two integers n, k, the number of move and the layers of guava tower. The next line contains n integers where the i-th integer p_i is the size of disk you should move at i-th step.

- $1 \leq T \leq 100$
- $1 \leq n \leq 10^6$
- $1 \leq k < 20$
- $1 \leq p_i \leq k$

Output

For each test case, output `Yes` if the sequence is a solution. Otherwise, output `No`.

Sample Input

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

Sample Output

```
Yes
No
No
```

Hungry TWT

Description

Hungry TWT is hungry again!

The Hungry TWT is a strange creature with its face looking like TWT. At the first day of each year, it will wakes up and starts to find some food as the only meal that year. It has a good appetite and it will eat anything it wants. Furthermore, due to its destructive power, there's nothing can stop Hungry TWT. As a result, people living in TMT Kingdom will try to hide themselves in the safest place they can ever find at the beginning of each year.

However, while the population grows over years, it becomes harder and harder to find enough space to hide all the people. In fact, thousands of people has been eaten by Hungry TWT in the past ten year. Hence the king of TMT Kingdom, Tomato, decides to start a campaign against Hungry TWT.

Sadly, and unsurprisingly, all those soldiers Tomato sent are eaten. So, instead of fight against Hungry TWT, Tomato decides to prepare a lot of meals for it. And prays that Hungry TWT will eat those meals. Fortunately, Hungry TWT is very clever and find the meals Tomato prepared much more delicious and easier to get. So, it left all the people it find.

The story starts here.

Hungry TWT discover that the order of eating those meals does affect its satisfaction level. Since foods will get cold or even rot, it must decides the order in a optimal way to maximize its satisfaction level. Can Hungry TWT find out the best order?

Input

The first line contains an integer T indicating the total number of test cases. Each test case starts with a line containing four integers N, M, K, D, the number of cities in TMT kingdom, the number or road in TMT kingdom, the number of meals Tomato prepares, satisfaction decrease factor. The following line contains K integers $L_1, L_2, \cdots, L_K$, the location of each meal. The third line contains K integers $V_1, V_2, \cdots, V_K$ the deliciousness of each meal. The next M lines contains three integers each, A_i, B_i, C_i, the two cities this road connect and the time needed to walk through it.

- $1 \leq T \leq 20$
- $1 \leq N \leq 100$
- $1 \leq M \leq 10000$
- $1 \leq K \leq 20$
- $1 \leq D \leq 10^6$
- $0 \leq A_i, B_i, L_i < N$
- $0 \leq V_i \leq 10^7$
- $0 \leq C_i \leq 10^6$

Hungry TWT is located in city 0 at time 0. If Hungry TWT eat a meal with deliciousness v at time t as the i-th meal it eat, it'll get $v - t - (i-1) \times D$ unit of satisfaction. And the satisfaction level is the summation of satisfaction of all meals Hungry TWT ate.

All the cities in TMT kingdom is connected either directly or indirectly, and you may assume it takes 0 unit of time for Hungry TWT to finish a meal.

Output

Please output the satisfaction level of the optimal plan. Note that Hungry TWT doesn't need to eat all the meals.

Sample Input

```
1
3 3 2 1
1 2
5 5
0 1 1
1 2 1
0 2 2
```

Sample Output

```
6
```

Instant Noodles

Description

Instant noodles are so important for coders, its importance is just as same as cokes and pizza. Now its near midnight, and you want to have some instant noodles. There are n types of instant noodles that you can have, and you want to eat in total B grams from these.

Different type of instant noodles have different flavor, and you do want to stay healthy. So you decided to eat them evenly and carefully. More specifically, you set two bounds L and U, and a magic integer (the container volume that you can grab them onto your scale) k, such that for any k types of instant noodles (you can pick them arbitrarily), the total weight that you eat will be between $\frac{Lk}{n}B$ and $\frac{Uk}{n}B$, inclusively.

But, there are definitely some kind of instant noodles that you like, and some you dislike. For each type of instant noodle i, you gave a lovely coefficient w_i to it. That is, if you eat t grams of i-type, then you will gain a lovely value $w_i \cdot t$.

The goal tonight is to find the maximum total lovely value that you could have. Hurry up! Otherwise the instant noodles will become cold and not delicious.

Input

The first line contains an integer T indicating the total number of test cases. Each test case starts with a line containing two integers n, k and three real numbers B, L, U. The following line contains n real numbers $w_1, w_2, \cdots, w_n$. There are at most four digits after the decimal to each number in the input.

- $1 \leq T \leq 50$
- $1 \leq n \leq 100$
- $1 \leq k \leq n$
- $1 \leq B \leq 100.0$
- $0 \leq L \leq 1.0 \leq U \leq 10.0$
- $0 \leq w_i \leq 100.0$

Output

For each test case please output the maximum value described above. The answer will be considered correct if the absolute or relative error does not exceed 10^{-6}.

Sample Input

```
3
3 1 10.0 0.9 1.1
1 2 3
3 2 10.0 0.9 1.1
1 1 1
3 3 10.0 0.9 1.1
1 1 100.0
```

Sample Output

```
20.666667
10.000000
1000.000000
```

Just Composite

Description

Solve it, or loss it

Shik is a prime lover. He knows every theorem of prime, every algorithm related to prime and even every prime used in published paper. As a rival of Shik in the NTU-ACM club, you decide to become a master of composite number.

After asking Fcrh, you plan to learn things about **compositeness** as the first step to master composite number. As a result, you design a problem about compositeness:

Give you several integers, find the maximum compositeness among all non-empty subsets.

The compositenes of a (multi-)set of integers is defined as the number of pair that's not coprime divided by the size of set.

Input

The first line contains an integer T indicating the total number of test cases. Each test case starts with a line containing an integer n, the number of integers. The next line contains n integers, $c_1, c_2, c_3, \ldots, c_n$.

- $1 \leq T \leq 100$
- $1 \leq n \leq 200$
- $1 \leq c_i \leq 10^{18}$

Output

For each test case, output the maximum compositeness among all subsets in the fraction form p/q with $\gcd(p, q) = 1$ and $q > 0$.

Sample Input

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

Sample Output

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