The problem statement has recently been changed. View the changes.

A. Zhan's Blender

1 second, 256 megabytes

Today, a club fair was held at "NSPhM". In order to advertise his pastry club, Zhan decided to demonstrate the power of his blender.

To demonstrate the power of his blender, Zhan has n fruits.

The blender can mix up to x fruits per second.

In each second, Zhan can put up to y fruits into the blender. After that, the blender will blend $\min(x, c)$ fruits, where c is the number of fruits inside the blender. After blending, blended fruits are removed from the blender.

Help Zhan determine the minimum amount of time required for Zhan to blend all fruits.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 1000$). The description of the test cases follows.

The first line of each test case contains one integer n ($0 \le n \le 10^9$) — the number of fruits Zhan has.

The second line of each test case contains two integers x and y ($1 \le x, y \le 10^9$) — the number of fruits the blender can blend per second and the number of fruits Zhan can put into the blender per second.

Output

For each testcase, output a single integer — the minimum number of seconds to blend all fruits.

input

```
5
5
3 4
3
1 2
6
4 3
100
4 3
9
3 3
```

output

```
2
3
2
34
3
```

In the first example, you can first put 2 fruits in the blender. After that, the blender will mix these 2 fruits, and in the end, there will be 0 fruits left in the blender. Then you can put 3 fruits into the blender, after which the blender will mix these 3 fruits.

In the second example, you can put 1 fruit into the blender 3 times.

In the third example, you can first put 3 fruits into the blender, then add another 3 fruits.

B. Battle for Survive

1 second, 256 megabytes

Eralim, being the mafia boss, manages a group of n fighters. Fighter i has a rating of a_i.

Eralim arranges a tournament of $n - 1$ battles, in each of which two not yet eliminated fighters i and j ($1 \le i < j \le n$) are chosen, and as a result of the battle, fighter i is eliminated from the tournament, and the rating of fighter j is reduced by the rating of fighter i. That is, a_j is decreased by a_i. Note that fighter j's rating can become negative. The fighters indexes do not change.

Eralim wants to know what maximum rating the last remaining fighter can preserve if he chooses the battles optimally.

Input

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

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

The second line of each test case contains n integers $a_1, a_2, \ldots a_n$ ($1 \le a_i \le 10^9$) — the ratings of the fighters.

The sum of n over all testcases does not exceed $2 \cdot 10^5$.

Output

For each testcase, output a single integer — the maximum rating that the last remaining fighter can preserve.

input

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

output

```
-1
8
2
7
8
```

In the first example, you can arrange a fight between fighters with indices 1 and 2, where the fighter with index 2 will win. The rating of the last fighter, that is, the fighter with index 2, will be $1 - 2 = -1$.

In the second example, you can first conduct a fight between fighters with indices 1 and 2, where the fighter with index 2 will win, and then conduct a fight between fighters with indices 2 and 3, where the fighter with index 3 will win.

The rating of the fighter with index 2 after the first fight will be $2 - 2 = 0$. The rating of the fighter with index 3 after the second fight will be $8 - 0 = 8$.

C. Password Cracking

2 seconds, 256 megabytes

Dimash learned that Mansur wrote something very unpleasant about him to a friend, so he decided to find out his password at all costs and discover what exactly he wrote.

Believing in the strength of his password, Mansur stated that his password — is a binary string of length n. He is also ready to answer Dimash's questions of the following type:

Dimash says a binary string t, and Mansur replies whether it is true that t is a substring of his password.

Help Dimash find out the password in no more than $2n$ operations; otherwise, Mansur will understand the trick and stop communicating with him.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 100$). The description of the test cases follows.

Interaction

At the beginning of each test case, first read n ($1 \le n \le 100$) — the size of the binary string. Then proceed to guessing it.

To guess each string s, you can make no more than $2n$ queries of the following type:

- "`? t`", where t is a binary string such that ($1 \le |t| \le n$).

In response to this query, you will receive 1 if t is a substring of s, and 0 otherwise.

Once you receive the answer, output a single string in the following format:

- "`! s`", where s is a binary string of size n.

After that, proceed to solve the next test case.

If you make an incorrect attempt or exceed the limit of $2n$ attempts, you will receive -1 instead of an answer and get the verdict `Wrong answer`. In this case, your program should terminate immediately to avoid undefined verdicts.

After outputting the queries, do not forget to output a newline character and flush the output buffer. Otherwise, you will receive the verdict `Solution "hung"`. To flush the buffer, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- refer to the documentation for other languages.

Hacks:

To use hacks, use the following format of tests:

The first line should contain a single integer t ($1 \le t \le 100$) — the number of test cases.

The first line of each test case should contain a single number n ($1 \le n \le 100$) — the length of the string. The second line should contain a binary string of size n.

input

```
4
3

0

0

1

4

4

2
```

output

```

? 00

? 000

? 010

! 010

! 1100

! 0110

! 10
```

In the first example, the string 010 is given. Therefore, the answers to the queries are as follows:

"`? 00`" 00 is not a substring of 010, so the answer is 0.

"`? 000`" 000 is not a substring, so the answer is 0.

"`? 010`" 010 is a substring, so the answer is 1.

In the second example, the string is 1100, in the third 0110, and in the fourth 10.

D. Minimize the Difference

2 seconds, 256 megabytes

Zhan, tired after the contest, gave the only task that he did not solve during the contest to his friend, Sungat. However, he could not solve it either, so we ask you to try to solve this problem.

You are given an array $a_1, a_2, \ldots, a_n$ of length n. We can perform any number (possibly, zero) of operations on the array.

In one operation, we choose a position i ($1 \le i \le n-1$) and perform the following action:

- $a_i := a_i - 1$, and $a_{i+1} := a_{i+1} + 1$.

Find the minimum possible value of $\max(a_1, a_2, \ldots, a_n) - \min(a_1, a_2, \ldots, a_n)$.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^5$). The description of the test cases follows.

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

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^{12}$).

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 minimum possible value of $\max(a_1, a_2, \ldots, a_n) - \min(a_1, a_2, \ldots, a_n)$.

input

```
5
1
1
3
1 2 3
4
4 1 2 3
4
4 2 3 1
5
5 14 4 10 2
```

output

```
0
2
1
1
3
```

In the third testcase, you can perform the operation twice with $i = 1$.

After that, the array is $a = [2, 3, 2, 3]$, and $\max(2, 3, 2, 3) - \min(2, 3, 2, 3) = 3 - 2 = 1$.

E. Prefix GCD

2 seconds, 256 megabytes

Since Mansur is tired of making legends, there will be no legends for this task.

You are given an array of positive integer numbers $a_1, a_2, \ldots, a_n$. The elements of the array can be rearranged in any order. You need to find the smallest possible value of the expression

$$\gcd(a_1) + \gcd(a_1, a_2) + \ldots + \gcd(a_1, a_2, \ldots, a_n),$$

where $\gcd(a_1, a_2, \ldots, a_n)$ denotes the greatest common divisor (GCD) of $a_1, a_2, \ldots, a_n$.

Input

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

The first line of each test case contains a single number n ($1 \le n \le 10^5$) — the size of the array.

The second line of each test case contains n numbers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^5$) — the initial array.

The sum of n over all test cases does not exceed 10^5.

The sum of $\max(a_1, a_2, \ldots, a_n)$ over all test cases does not exceed 10^5.

Output

For each test case, output a single number on a separate line — the answer to the problem.

input

```
5
3
4 2 2
2
6 3
3
10 15 6
5
6 42 12 52 20
4
42 154 231 66
```

output

```
6
6
9
14
51
```

In the first test case, the elements can be rearranged as follows: $[2, 4, 2]$. Then the answer will be $\gcd(2) + \gcd(2, 4) + \gcd(2, 4, 2) = 2 + 2 + 2 = 6$.

In the third test case, the elements can be rearranged as follows: $[6, 10, 15]$. Then the answer will be $\gcd(6) + \gcd(6, 10) + \gcd(6, 10, 15) = 6 + 2 + 1 = 9$.

The problem statement has recently been changed. View the changes. ×

F1. Game in Tree (Easy Version)

4 seconds, 256 megabytes

This is the easy version of the problem. In this version, $\mathbf{u = v}$. You can make hacks only if both versions of the problem are solved.

Alice and Bob are playing a fun game on a tree. This game is played on a tree with n vertices, numbered from 1 to n. Recall that a tree with n vertices is an undirected connected graph with $n - 1$ edges.

Alice and Bob take turns, with Alice going first. Each player starts at some vertex.

On their turn, a player must move from the current vertex to a neighboring vertex that has not yet been visited by anyone. The first player who cannot make a move loses.

You are given two vertices u and v. Represent the simple path from vertex u to v as an array $p_1, p_2, p_3, \ldots, p_m$, where $p_1 = u$, $p_m = v$, and there is an edge between p_i and p_{i+1} for all i ($1 \le i < m$).

You need to determine the winner of the game if Alice starts at vertex 1 and Bob starts at vertex p_j for each j (where $1 \le j \le m$).

Input

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

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

Each of the following $n - 1$ lines contains two integers a and b ($1 \le a, b \le n$), denoting an undirected edge between vertices a and b. It is guaranteed that these edges form a tree.

The last line of each test case contains two integers u and v ($2 \le u, v \le n$, $\mathbf{u = v}$).

It is guaranteed that the path from u to v does not pass through vertex 1.

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 m lines.

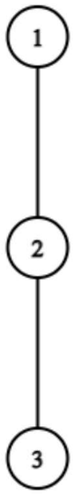
In the i-th line, print the winner of the game if Alice starts at vertex 1 and Bob starts at vertex p_i. Print "`Alice`" (without quotes) if Alice wins, or "`Bob`" (without quotes) otherwise.

input

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

output

```
Bob
Alice
Alice
```



Tree from the first and second examples.

In the first test case, the path will be $(2, 2)$. Bob starts at vertex 2, Alice will not be able to move anywhere on her first turn and will lose.

In the second test case, the path will be $(3, 3)$. Bob starts at vertex 3, Alice will move to vertex 2, and Bob will have no remaining vertices to visit and will lose.

The problem statement has recently been changed. View the changes. ×

F2. Game in Tree (Hard Version)

4 seconds, 256 megabytes

This is the hard version of the problem. In this version, it is not guaranteed that $u = v$. You can make hacks only if both versions of the problem are solved.

Alice and Bob are playing a fun game on a tree. This game is played on a tree with n vertices, numbered from 1 to n. Recall that a tree with n vertices is an undirected connected graph with $n - 1$ edges.

Alice and Bob take turns, with Alice going first. Each player starts at some vertex.

On their turn, a player must move from the current vertex to a neighboring vertex that has not yet been visited by anyone. The first player who cannot make a move loses.

You are given two vertices u and v. Represent the simple path from vertex u to v as an array $p_1, p_2, p_3, \ldots, p_m$, where $p_1 = u$, $p_m = v$, and there is an edge between p_i and p_{i+1} for all i $(1 \le i < m)$.

You need to determine the winner of the game if Alice starts at vertex 1 and Bob starts at vertex p_j for each j (where $1 \le j \le m$).

Input

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

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

Each of the following $n - 1$ lines contains two integers a and b ($1 \le a, b \le n$), denoting an undirected edge between vertices a and b. It is guaranteed that these edges form a tree.

The last line of each test case contains two integers u and v ($2 \le u, v \le n$).

It is guaranteed that the path from u to v does not pass through vertex 1.

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 m lines.

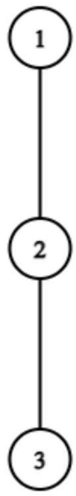
In the i-th line, print the winner of the game if Alice starts at vertex 1 and Bob starts at vertex p_i. Print "`Alice`" (without quotes) if Alice wins, or "`Bob`" (without quotes) otherwise.

input

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

output

```
Bob
Alice
Alice
Bob
Alice
Bob
Alice
```



Tree from the first example.

In the first test case, the path will be $(2, 3)$. If Bob starts at vertex 2, Alice will not be able to move anywhere on her first turn and will lose.

However, if Bob starts at vertex 3, Alice will move to vertex 2, and Bob will have no remaining vertices to visit and will lose.