**Problem 1: 30Marks**

There are 100 houses located on a **straight line**. The first house is numbered 1 and the last one is numbered 100. Some **M** houses out of these 100 are occupied by cops.

Thief Devu has just stolen PeePee's bag and is looking for a house to hide in.

PeePee uses fast 4G Internet and sends the message to all the cops that a thief named Devu has just stolen her bag and ran into some house.

Devu knows that the cops run at a maximum speed of **x** houses per minute in a straight line and they will search for a maximum of **y** minutes. Devu wants to know how many houses are safe for him to escape from the cops. Help him in getting this information.

**Input**

First line contains **T**, the number of test cases to follow.

First line of each test case contains 3 space separated integers: **M**, **x** and **y**.

For each test case, the second line contains **M** space separated integers which represent the house numbers where the cops are residing.

**Output**

For each test case, output a single line containing the number of houses which are safe to hide from cops.

**Constraints**

* 1 ≤ **T** ≤ 104
* 1 ≤ **x, y, M** ≤ 10

**Example**

**Input:**

3

4 7 8

12 52 56 8

2 10 2

21 75

2 5 8

10 51

**Output:**

0

18

9

**Explanation**

**Example 1 :** Cops in house 12 can cover houses 1 to 68, and cops in house 52 can cover the rest of the houses. So, there is no safe house.

**Example 2 :** Cops in house 21 can cover houses 1 to 41, and cops in house 75 can cover houses 55 to 95, leaving houses numbered 42 to 54, and 96 to 100 safe. So, in total 18 houses are safe.

**Problem 2: 30Marks**

Forgotten languages (also known as extinct languages) are languages that are no longer in use. Such languages were, probably, widely used before and no one could have ever imagined that they will become extinct at some point. Unfortunately, that is what happened to them. On the happy side of things, a language may be dead, but some of its words may continue to be used in other languages.

Using something called as *the Internet*, you have acquired a dictionary of **N** words of a forgotten language. Meanwhile, you also know **K** phrases used in modern languages. For each of the words of the forgotten language, your task is to determine whether the word is still in use in any of these **K** modern phrases or not.

**Input**

The first line of the input contains an integer **T** denoting the number of test cases. The description of **T** test cases follows.

The first line of a test case description contains two space separated positive integers **N**and **K**.

The second line of the description contains **N** strings denoting a dictionary of the forgotten language.

Each of the next **K** lines of the description starts with one positive integer **L** denoting the number of words in the corresponding phrase in modern languages. The integer is followed by **L** strings (not necessarily distinct) denoting the phrase.

**Output**

For each test case, output a single line containing **N** tokens (space-separated): if the **i**thword of the dictionary exists in at least one phrase in modern languages, then you should output **YES** as the **i**th token, otherwise **NO**.

**Constraints**

* 1 ≤ **T** ≤ 20
* 1 ≤ **N** ≤ 100
* 1 ≤ **K, L** ≤ 50
* 1 ≤ length of any string in the input ≤ 5

**Example**

**Input:**

2

3 2

piygu ezyfo rzotm

1 piygu

6 tefwz tefwz piygu ezyfo tefwz piygu

4 1

kssdy tjzhy ljzym kegqz

4 kegqz kegqz kegqz vxvyj

**Output:**

YES YES NO

NO NO NO YES

**Problem 3: 40Marks**

Little chief has his own restaurant in the city. There are **N** workers there. Each worker has his own salary. The salary of the **i**-th worker equals to **Wi** (**i** = **1**, **2**, ..., **N**). Once, chief decided to equalize all workers, that is, he wants to make salaries of all workers to be equal. But for this goal he can use only one operation: choose some worker and increase by 1 salary of each worker, except the salary of the chosen worker. In other words, the chosen worker is the loser, who will be the only worker, whose salary will be not increased during this particular operation. But loser-worker can be different for different operations, of course. Chief can use this operation as many times as he wants. But he is a busy man. That's why he wants to minimize the total number of operations needed to equalize all workers. Your task is to find this number.

**Input**

The first line of the input contains an integer **T** denoting the number of test cases. The description of **T** test cases follows. The first line of each test case contains a single integer **N** denoting the number of workers. The second line contains **N** space-separated integers **W1**, **W2**, ..., **WN** denoting the salaries of the workers.

**Output**

For each test case, output a single line containing the minimum number of operations needed to equalize all workers.

**Constraints**

* **1** ≤ **T** ≤ **100**
* **1** ≤ **N** ≤ **100**
* **0** ≤ **Wi** ≤ **10000** (**104**)

**Example**

**Input:**

2

3

1 2 3

2

42 42

**Output:**

3

0

**Explanation**

**Example Case 1.** Chief can equalize all salaries in 3 turns:

|  |  |  |
| --- | --- | --- |
| Turn ID | IDs of involved workers | Salaries after the move |
| 1 | 1 2 | 2 3 3 |
| 2 | 1 2 | 3 4 3 |
| 3 | 1 3 | 4 4 4 |

**\*Example Case 2.** All salaries are already equal. He doesn't need to do anything.