## A. Finding Sasuke

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

Naruto has sneaked into the Orochimaru's lair and is now looking for Sasuke. There are $T$ rooms there. Every room has a door into it, each door can be described by the number $n$ of seals on it and their integer energies $a_1$, $a_2$, ..., $a_n$. All energies $a_i$ are **nonzero** and do not exceed $100$ by absolute value. Also, $n$ **is even**.

In order to open a door, Naruto must find such $n$ seals with integer energies $b_1$, $b_2$, ..., $b_n$ that the following equality holds: $a_1 \cdot b_1 + a_2 \cdot b_2 + ... + a_n \cdot b_n = 0$. All $b_i$ must **be nonzero** as well as $a_i$ are, and also **must not exceed** $100$ by absolute value. Please find required seals for every room there.

### Input

The first line contains the only integer $T$ ($1 \le T \le 1000$) standing for the number of rooms in the Orochimaru's lair. The other lines contain descriptions of the doors.

Each description starts with the line containing the only even integer $n$ ($2 \le n \le 100$) denoting the number of seals.

The following line contains the space separated sequence of nonzero integers $a_1$, $a_2$, ..., $a_n$ ($|a_i| \le 100$, $a_i \ne 0$) denoting the energies of seals.

### Output

For each door print a space separated sequence of nonzero integers $b_1$, $b_2$, ..., $b_n$ ($|b_i| \le 100$, $b_i \ne 0$) denoting the seals that can open the door. If there are multiple valid answers, print any. It can be proven that at least one answer always exists.

#### Example

input

```
2
2
1 100
4
1 2 3 6
```

output

```
-100 1
1 1 1 -1
```

### Note

For the first door Naruto can use energies $[-100, 1]$. The required equality does indeed hold: $1 \cdot (-100) + 100 \cdot 1 = 0$.

For the second door Naruto can use, for example, energies $[1, 1, 1, -1]$. The required equality also holds: $1 \cdot 1 + 2 \cdot 1 + 3 \cdot 1 + 6 \cdot (-1) = 0$.

## B. A New Technique

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

All techniques in the ninja world consist of hand seals. At the moment Naruto is learning a new technique, which consists of $n \cdot m$ different seals, denoted by distinct numbers. All of them were written in an $n \times m$ table.

The table is lost now. Naruto managed to remember elements of each row from left to right, and elements of each column from top to bottom, but he doesn't remember the order of rows and columns. Please restore the table consistent with this data so that Naruto will be able to learn the new technique.

### Input

The first line of the input contains the only integer $t$ ($1 \le t \le 100\,000$) denoting the number of test cases. Their descriptions follow.

The first line of each test case description consists of two space-separated integers $n$ and $m$ ($1 \le n, m \le 500$) standing for the number of rows and columns in the table, respectively. All hand seals are encoded by the positive integers from $1$ to $n \cdot m$.

The following $n$ lines contain $m$ space separated integers each, denoting elements of an arbitrary row in the table left to right.

The following $m$ lines contain $n$ space separated integers each, denoting elements of an arbitrary column in the table top to bottom.

Sum of $nm$ over all test cases does not exceed $250\,000$. It is guaranteed that each row occurs in the input exactly once, as well as each column. It is also guaranteed that each number from $1$ to $nm$ occurs exactly once in all rows, as well as in all columns. Finally, it is guaranteed that a table consistent with the input exists.

### Output

For each test case, output $n$ lines with $m$ space-separated integers each, denoting the restored table. One can show that the answer is always unique.

#### Example

**input**

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

**output**

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

### Note

Consider the first test case. The matrix is $2 \times 3$. You are given the rows and columns in arbitrary order.

One of the rows is $[6, 5, 4]$. One of the rows is $[1, 2, 3]$.

One of the columns is $[1, 6]$. One of the columns is $[2, 5]$. One of the columns is $[3, 4]$.

You are to reconstruct the matrix. The answer is given in the output.

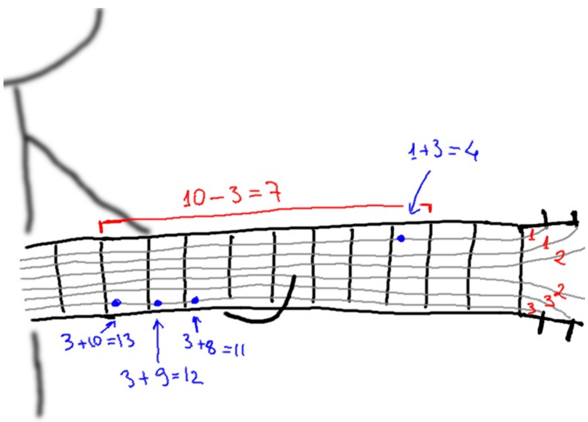
## C. Perform Easily

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

After battling Shikamaru, Tayuya decided that her flute is too predictable, and replaced it with a guitar. The guitar has $6$ strings and an infinite number of frets numbered from $1$. Fretting the fret number $j$ on the $i$-th string produces the note $a_i + j$.

Tayuya wants to play a melody of $n$ notes. Each note can be played on different string-fret combination. The easiness of performance depends on the difference between the maximal and the minimal indices of used frets. The less this difference is, the easier it is to perform the technique. Please determine the minimal possible difference.

For example, if $a = [1, 1, 2, 2, 3, 3]$, and the sequence of notes is $4, 11, 11, 12, 12, 13, 13$ (corresponding to the second example), we can play the first note on the first string, and all the other notes on the sixth string. Then the maximal fret will be $10$, the minimal one will be $3$, and the answer is $10 - 3 = 7$, as shown on the picture.




### Input

The first line contains $6$ space-separated numbers $a_1$, $a_2$, ..., $a_6$ ($1 \leq a_i \leq 10^9$) which describe the Tayuya's strings.

The second line contains the only integer $n$ ($1 \leq n \leq 100\,000$) standing for the number of notes in the melody.

The third line consists of $n$ integers $b_1$, $b_2$, ..., $b_n$ ($1 \leq b_i \leq 10^9$), separated by space. They describe the notes to be played. It's guaranteed that $b_i > a_j$ for all $1 \leq i \leq n$ and $1 \leq j \leq 6$, in other words, you can play each note on any string.

## Output

Print the minimal possible difference of the maximal and the minimal indices of used frets.

### Examples

**input**

```
1 4 100 10 30 5
6
101 104 105 110 130 200
```

**output**

```
0
```

**input**

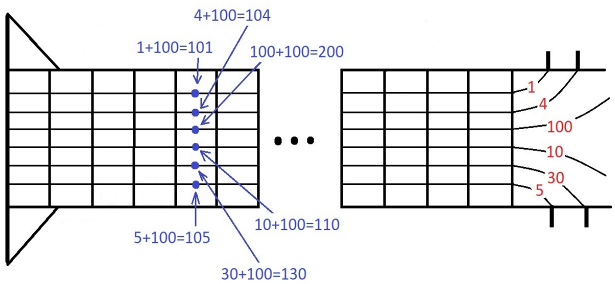
```
1 1 2 2 3 3
7
13 4 11 12 11 13 12
```

**output**

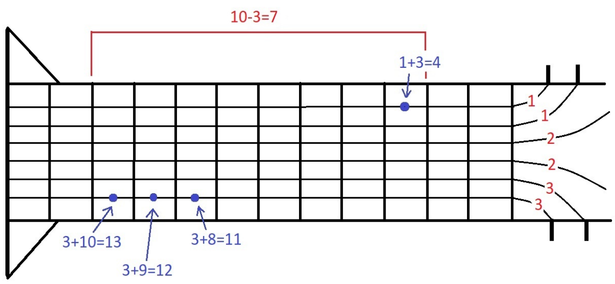
```
7
```

## Note

In the first sample test it is optimal to play the first note on the first string, the second note on the second string, the third note on the sixth string, the fourth note on the fourth string, the fifth note on the fifth string, and the sixth note on the third string. In this case the $100$-th fret is used each time, so the difference is $100 - 100 = 0$.




In the second test it's optimal, for example, to play the second note on the first string, and all the other notes on the sixth string. Then the maximal fret will be $10$, the minimal one will be $3$, and the answer is $10 - 3 = 7$.




# D. Shurikens

time limit per test: 1 second

memory limit per test: 256 megabytes

input: standard input

output: standard output

Tenten runs a weapon shop for ninjas. Today she is willing to sell $n$ shurikens which cost $1$, $2$, ..., $n$ ryo (local currency). During a day, Tenten will place the shurikens onto the showcase, which is empty at the beginning of the day. Her job is fairly simple: sometimes Tenten places another shuriken (from the available shurikens) on the showcase, and sometimes a ninja comes in and buys a shuriken from the showcase. Since ninjas are thrifty, they always buy the **cheapest** shuriken from the showcase.

Tenten keeps a record for all events, and she ends up with a list of the following types of records:

- + means that she placed another shuriken on the showcase;
- - x means that the shuriken of price $x$ was bought.

Today was a lucky day, and all shurikens were bought. Now Tenten wonders if her list is consistent, and what could be a possible order of placing the shurikens on the showcase. Help her to find this out!

## Input

The first line contains the only integer $n$ ($1 \leq n \leq 10^5$) standing for the number of shurikens.

The following $2n$ lines describe the events in the format described above. It's guaranteed that there are exactly $n$ events of the first type, and each price from $1$ to $n$ occurs exactly once in the events of the second type.

### Output

If the list is consistent, print "YES". Otherwise (that is, if the list is contradictory and there is no valid order of shurikens placement), print "NO".

In the first case the second line must contain $n$ space-separated integers denoting the prices of shurikens in order they were placed. If there are multiple answers, print any.

#### Examples

**input**

```
4
+
+
- 2
+
- 3
+
- 1
- 4
```

**output**

```
YES
4 2 3 1
```

**input**

```
1
- 1
+
```

**output**

```
NO
```

**input**

```
3
+
+
+
- 2
- 1
- 3
```

**output**

```
NO
```

### Note

In the first example Tenten first placed shurikens with prices $4$ and $2$. After this a customer came in and bought the cheapest shuriken which costed $2$. Next, Tenten added a shuriken with price $3$ on the showcase to the already placed $4$-ryo. Then a new customer bought this $3$-ryo shuriken. After this she added a $1$-ryo shuriken. Finally, the last two customers bought shurikens $1$ and $4$, respectively. Note that the order $[2, 4, 3, 1]$ is also valid.

In the second example the first customer bought a shuriken before anything was placed, which is clearly impossible.

In the third example Tenten put all her shurikens onto the showcase, after which a customer came in and bought a shuriken with price $2$. This is impossible since the shuriken was not the cheapest, we know that the $1$-ryo shuriken was also there.

## E. Solo mid Oracle

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

Meka-Naruto plays a computer game. His character has the following ability: given an enemy hero, deal $a$ instant damage to him, and then heal that enemy $b$ health points at the end of every second, for exactly $c$ seconds, starting one second after the ability is used. That means that if the ability is used at time $t$, the enemy's health decreases by $a$ at time $t$, and then increases by $b$ at time points $t+1$, $t+2$, ..., $t+c$ due to this ability.

The ability has a cooldown of $d$ seconds, i. e. if Meka-Naruto uses it at time moment $t$, next time he can use it is the time $t+d$. Please note that he can only use the ability at integer points in time, so all changes to the enemy's health also occur at integer times only.

The effects from different uses of the ability may stack with each other; that is, the enemy which is currently under $k$ spells gets $k \cdot b$ amount of heal this time. Also, if several health changes occur at the same moment, they are all counted at once.

Now Meka-Naruto wonders if he can kill the enemy by just using the ability each time he can (that is, every $d$ seconds). The enemy is killed if their health points become $0$ or less. Assume that the enemy's health is not affected in any way other than by Meka-Naruto's character ability. What is the maximal number of health points the enemy can have so that Meka-Naruto is able to kill them?

### Input

The first line contains an integer $t$ ($1 \le t \le 10^5$) standing for the number of testcases.

Each test case is described with one line containing four numbers $a$, $b$, $c$ and $d$ ($1 \le a, b, c, d \le 10^6$) denoting the amount of instant damage, the amount of heal per second, the number of heals and the ability cooldown, respectively.

### Output

For each testcase in a separate line print $-1$ if the skill can kill an enemy hero with an arbitrary number of health points, otherwise print the maximal number of health points of the enemy that can be killed.

#### Example

**input**

```
7
1 1 1 1
2 2 2 2
1 2 3 4
4 3 2 1
228 21 11 3
239 21 11 3
1000000 1 1000000 1
```

**output**

```
1
2
1
5
534
-1
500000500000
```

### Note

In the first test case of the example each unit of damage is cancelled in a second, so Meka-Naruto cannot deal more than 1 damage.

In the fourth test case of the example the enemy gets:

- $4$ damage ($1$-st spell cast) at time $0$;
- $4$ damage ($2$-nd spell cast) and $3$ heal ($1$-st spell cast) at time $1$ (the total of $5$ damage to the initial health);
- $4$ damage ($3$-nd spell cast) and $6$ heal ($1$-st and $2$-nd spell casts) at time $2$ (the total of $3$ damage to the initial health);
- and so on.

One can prove that there is no time where the enemy gets the total of $6$ damage or more, so the answer is $5$. Please note how the health is recalculated: for example, $8$-health enemy would **not** die at time $1$, as if we first subtracted $4$ damage from his health and then considered him dead, before adding $3$ heal.

In the sixth test case an arbitrarily healthy enemy can be killed in a sufficient amount of time.

In the seventh test case the answer does not fit into a 32-bit integer type.