NCTU Competitive Programming I

2020 Spring

Midterm Exam 1

ID	Problem Name	Time Limit
A	Awesome Arrangement	2 sec
B	Brilliant Barbeque	10 sec
C	Cursed Code	5 sec
D	Diamond Discovery	10 sec
E	Eradicative Expressions	2 sec
F	Frontend Failure	2 sec
G	Geometric Greatness	2 sec
H	Hall of Hyper-chairs™	3 sec

NCTU Competitive Programming I
2020 Spring
Midterm Exam 1

Rules

Any violation of the rules is considered cheating and will result in a disqualification.

1. You should take place in the exam using your personal and only DOMjudge account given prior to the exam.
2. You must not discuss problems or share ideas/solutions with any other people.
3. Should you have a question about the problems or face any exam-related issue, please use the "request clarification" feature of DOMjudge. Do not ask or discuss with any other people.
4. Announcements made during the exam can be viewed using the "Clarifications" interface of DOMjudge.
5. You may use existing source code as parts of your solution. However, the code you use must either be in the lecture notes or be written by you prior to the exam.
6. You may search for and read documents or references online, but copying solution codes are not allowed.
7. Any malicious action interfering the exam is prohibited.

Scoring

1. You must submit your solutions via DOMjudge. The judge system will only respond to submissions that are submitted within the exam duration (300 minutes). The response to each run must be one of the following:
   - `Correct`: The judge accepts your code.
   - `Compiler-Error`: Your code cannot be successfully compiled.
   - `TimeLimit`: Your program consumes too much time.
   - `Run-Error`: Your program terminates with an non-zero return code, which often means your program is terminated by the operating system.
   - `Wrong-Answer`: The judge rejects the output of your program.
   - `No-Output`: Your program does not generate any output.

2. The score you get for this exam is based on the number of problems to which a correct solution is submitted:

Solved	Score
0	20
1	30
2	35
3	39
4	43
5	46
6	48
7	49
8	50

Hint

The problems are sorted by their names (lexicographically) and not by increasing difficulty; problems that appear first are not necessarily easier. Because of this, it is recommended that you read every problem.

Almost blank page

Problem A
Awesome Arrangement

Time limit: 2 seconds

Memory limit: 512 megabytes

Problem Description

Saiki Kusuo is a high school student with all kinds of psychic powers. One of his most useful abilities is Apport (アポート), which can be used to teleport an object. To do that, he has to first select another object near it and then have the objects swap positions with each other.

Kusuo's favorite food is coffee jellies. There are n pieces of coffee jelly in a row. For each $1 \leq i \leq n$, the i^{th} jelly is initially placed at position i and has value a_i. All of their values are different. Since Kusuo is a perfectionist, he hopes that the jellies are in a "perfect order": for each $1 \leq i \leq n$, the jelly at position i should have value b_i instead. The list $\langle b_i \rangle$ is a permutation of $\langle a_i \rangle$.

To achieve that perfect order, Kusuo can perform some moves with Apport. For each move, he can choose an index i ($1 \leq i \leq n-1$) and swap the jellies at positions i and $i+1$. Whenever he swaps two jellies with values x and y, he has to pay a mana cost $(x+y) \mod k$. As mana is pretty valuable to him (though not more than coffee jellies), he wants to pay as less total mana cost as possible. What is the minimum cost?

Input Format

The first line contains two integers n and k. The second line contains n space-separated integers $a_1, a_2, \ldots, a_n$. The third line contains n space-separated integers $b_1, b_2, \ldots, b_n$.

Output Format

Output an integer denoting the minimum mana cost required to achieve the perfect order.

Technical Specification

- $2 \leq n \leq 10^5$
- $2 \leq k \leq 10$
- $0 \leq a_i, b_i \leq 10^9$
- $\langle a_i \rangle$ and $\langle b_i \rangle$ are permutations of each other, and the values a_i are unique.

Sample Input 1

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

Sample Output 1

```
6
```

Almost blank page

Problem B
Brilliant Barbeque

Time limit: 10 seconds
Memory limit: 512 megabytes

Problem Description

Allen is hungry, and he has prepared 9 food items for the Barbeque. Due to limitations of the grill's size, he arranges his food into 3 rows and 3 items per row, i.e., a 3×3 grid. Noticing that the grill is not heated up uniformly, Allen decides to swap some food items' positions every minute so they can all be well-cooked faster. At the moment that all of them are well-cooked, they should also be placed at some specific positions on the grill.

Allen first puts his food items onto the grill in an arbitrary order. Then at the end of each minute, he can choose either two rows or two columns of food and swap them. If he is satisfied with the current food arrangement, he can also choose not to do anything.

Before a food item is put onto the grill, its temperature is 0°C, since Allen stores all his food in a fridge at 0°C. If something on the grill is placed somewhere where fire is more powerful, its temperature rises up faster. To describe how powerful the fire is, we define the "firepower" of a location as the increment of temperature (in Celsius degrees) of an item per minute if it is put there. For example, if a food item is put somewhere with a firepower of 5, its temperature increases by 5°C every minute, and if it's at somewhere with zero firepower, its temperature never rises and thus the item never gets cooked. We define the "cooked point" of a food item as the temperature at which it gets well-cooked.

Given the firepower of all 3×3 positions of the grill and the cooked points of 9 food items, help Allen to find out the minimum time required to well-cook all of them. Note that if an item has temperature higher than its cooked point, it will be overcooked and charred so Allen refuses to eat it. Furthermore, all of them should be well-cooked on the grill at the same time, at their required positions. The grill's fire is stable; in other words, the firepower of a position is constant. Lastly, the deadline of NA (Computer Network Administration, 計算機網路管理) is coming, so Allen is busy doing his homework. He will only glance at the barbeque every 60 seconds, so please make sure that the foods will be well-cooked at the end of some minute. Allen is too hungry to wait. If his meal cannot be prepared within 15 minutes, tell him that "`NA IS HARD, SO IS LIFE.`".

Input Format

The first line is an integer T indicating the number of test cases. Then T test cases follow. For each test case there are 2 sections, each of them contains 3 lines of 3 integers, representing a 3×3 grid.

The first section of each case describes the firepower map of the Barbeque: the j^{th} value on the i^{th} line corresponds to the i^{th} row, j^{th} column on the grill.

The second section of each case indicates the food items' cooked points and their required locations: the j^{th} value on the i^{th} line means that there is an item having that value as its cooked point and should be placed at the i^{th} row, j^{th} column.

Output Format

Output a single integer indicating the minimum time, in minutes, for all food items to be well-cooked. If it is impossible to meet the requirements or the required time is longer than 15 minutes, output "`NA IS HARD, SO IS LIFE.`" (without quotes).

Technical Specification

- $1 \leq T \leq 10$
- All firepower values are integers between 0 and 100 (inclusive).
- The cooked points of all food items are integers between 1 and 100 (inclusive).

Sample Input 1

```
4
1 1 0
1 1 0
1 1 0
1 1 2
1 1 2
1 1 2
1 1 1
1 2 1
1 1 1
1 1 1
1 1 1
1 1 1
4 4 4
4 4 4
4 4 4
2 2 2
2 2 2
2 2 2
7 1 3
4 0 7
3 1 4
30 44 33
19 42 43
19 42 28
```

Sample Output 1

```
2
NA IS HARD, SO IS LIFE.
NA IS HARD, SO IS LIFE.
10
```

Hint

In the first sample, there are two kinds of food items whose cooked points are 1 and 2 respectively. Let the former be beef (B) and the latter be sausage (S). At the beginning, Allen can choose to put sausages in the first column and beef in other ones. Here is a diagram describing the temperatures and positions of each item on the Barbeque grill initially:

S(0) B(0) B(0)
S(0) B(0) B(0)
S(0) B(0) B(0)

At the end of the first minute, Allen can swap the right and center columns. The diagram becomes:

S(1) B(0) B(1)
S(1) B(0) B(1)
S(1) B(0) B(1)

At the end of the second minute, Allen can swap the left and right columns. The diagram should be:

B(1) B(1) S(2)
B(1) B(1) S(2)
B(1) B(1) S(2)

Thus all food items are well-cooked and at their required locations. Allen must therefore spend 2 minutes to prepare his meal.

In the second sample, the item put on the center of the grill will be overcooked.

In the third sample, all the food will be well-cooked after 30 seconds, while Allen is debugging for his NA homework and not noticing his dinner is ready. Not until the end of the first minute will he find that his dinner have been burnt, and in this case, he will realize his life is hard.

Almost blank page

Problem C
Cursed Code

Time limit: 5 seconds
Memory limit: 512 megabytes

Problem Description

Carl is editing a code file with r lines and c characters in each line. Lines are indexed from top to bottom and characters are indexed from left to right, both starting from 0. The j^{th} character of the i^{th} line can be described as coordinates (i, j), where $0 \leq i < r$ and $0 \leq j < c$. Initially, each character is an `f`.

Carl uses the cursed editor "`miv`" to edit his code. The characters are arranged into an $r \times c$ grid inside the editor. There is a pointer that points to a character, initially at $(0, 0)$. A main feature of `miv` is the "pointer reset" operation. When it is applied, `miv` will select the `f` at (i, j) with minimum j and move the pointer there. If there are multiple `f`'s with minimum j, it will choose the one with minimum i among them.

One day, the editor suddenly gained consciousness itself and started destroying the file. The file destruction process is as follows:

1. `miv` does the "right shift" operation for k times. In each "right shift", the pointer is moved to the next `f` to its right. If there are no `f`'s on the right, it is moved to the leftmost `f` in that row instead.
2. The character currently pointed at by the pointer is erased and replaced by a space. After that, if all characters in that row are erased, `miv` performs a "pointer reset". Otherwise, it performs a "right shift".
3. `miv` does the "down shift" operation for k times. In each "down shift", the pointer is moved to the next `f` below it. If there are no `f`'s below, it is moved to the topmost `f` in that column instead.
4. The character currently pointed at by the pointer is erased and replaced by a space. After that, if all characters in that column are erased, `miv` performs a "pointer reset". Otherwise, it performs a "down shift".
5. The steps 1 to 4 are repeated until all characters are erased except one. Then a segmentation fault occurs, ending the whole process.

Since the process is done so fast, Carl only sees one character left after one blink. What are the coordinates of that character?

Input Format

The input contains three spaced-separated integers r, c, k in one line.

Output Format

Output two integers m, n separated by a space, meaning that the last character left is the n^{th} one on line m.

Technical Specification

- $1 \leq r \leq 500, 1 \leq c \leq 500$
- $1 \leq k \leq \min(r, c)$

Sample Input 1	Sample Output 1
`3 3 2`	`2 2`

Hint

In the sample, the code is originally 3×3 `f`'s. Let `[ ]` represent the pointer and `.` denote a space. After doing step 1, the pointer moves to $(0, 2)$:

```
[f] f  f        f [f] f        f  f [f]
 f  f  f   ->   f  f  f   ->   f  f  f
 f  f  f        f  f  f        f  f  f
```

After doing step 2, the character at $(0, 2)$ is erased and the pointer moves to $(0, 0)$:

```
 f  f [f]       f  f [.]      [f] f  .
 f  f  f   ->   f  f  f   ->   f  f  f
 f  f  f        f  f  f        f  f  f
```

After step 3, the pointer moves to $(2, 0)$:

```
[f] f  .        f  f  .        f  f  .
 f  f  f   ->  [f] f  f   ->   f  f  f
 f  f  f        f  f  f       [f] f  f
```

After step 4, the character at $(2, 0)$ is erased and the pointer moves to $(0, 0)$:

```
 f  f  .        f  f  .       [f] f  .
 f  f  f   ->   f  f  f   ->   f  f  f
[f] f  f       [.] f  f        .  f  f
```

This process continues until all characters are erased except one, and the final code look likes this:

```
 .  .  .
 .  .  .
 .  . [f]
```

so the answer is "`2 2`".

Problem D
Diamond Discovery

Time limit: 10 seconds
Memory limit: 512 megabytes

Problem Description

You've landed on a mysterious island somewhere in the world. Legend has it that on the island, there is a buried treasure consisting of stacks of diamonds. The island's terrain can be described as an $n \times m$ rectangular grid, where each cell is denoted by integer coordinates (x, y) where $1 \leq x \leq n$ and $1 \leq y \leq m$. The land in each cell has an uniform height, and you have a topographic map containing altitude information about the island: the cell at (i, j) has height $h_{i,j}$.

You begin your journey at (x_s, y_s) and the treasure is located at (x_t, y_t). You can move from the cell at (x_u, y_u) to the one at (x_v, y_v) under the following conditions:

1. $1 \leq x_u, x_v \leq n$ and $1 \leq y_u, y_v \leq m$ must hold.
2. $|x_u - x_v| + |y_u - y_v| = 1$ must hold.
3. You have brought a ladder of non-negative integer length to the island. If the height difference $|h_{x_u,y_u} - h_{x_v,y_v}| \leq l$, where l is the length of your ladder, then you can make the move. The ladder will be with you at all times.
4. You have brought k "Potions of Leaping" to the island. If you have at least one potion left, then you can consume one potion and make the move regardless of height difference.

Find the minimum length of the ladder to prepare so that you can reach the treasure.

Input Format

The first line contains the three space-separated integers n, m, k.

Then n lines follow, each containing m space-separated integers. They represent the topographic map; the j^{th} integer on the i^{th} line is the height $h_{i,j}$.

The last line contains the four space-separated integers x_s, y_s, x_t, y_t.

Output Format

Output an integer, the minimum length of the ladder you should prepare.

Technical Specification

- $1 \leq n, m \leq 10^5, 1 \leq n \times m \leq 10^5$
- $0 \leq k \leq 10^5$
- $1 \leq h_{i,j} \leq 10^9$

- $1 \leq x_s, x_t \leq n, 1 \leq y_s, y_t \leq m$
- $(x_s, y_s) \neq (x_t, y_t)$

Sample Input 1

```
3 3 0
1 1 7
7 4 8
6 9 7
1 1 3 3
```

Sample Output 1

```
3
```

Sample Input 2

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

Sample Output 2

```
1
```

Hint

In the first sample, preparing a ladder of length 3 and moving along the blue arrows is the optimal solution.

In the second sample, you can prepare a ladder of length 1 and use the potion to move from $(1, 2)$ to $(1, 3)$.

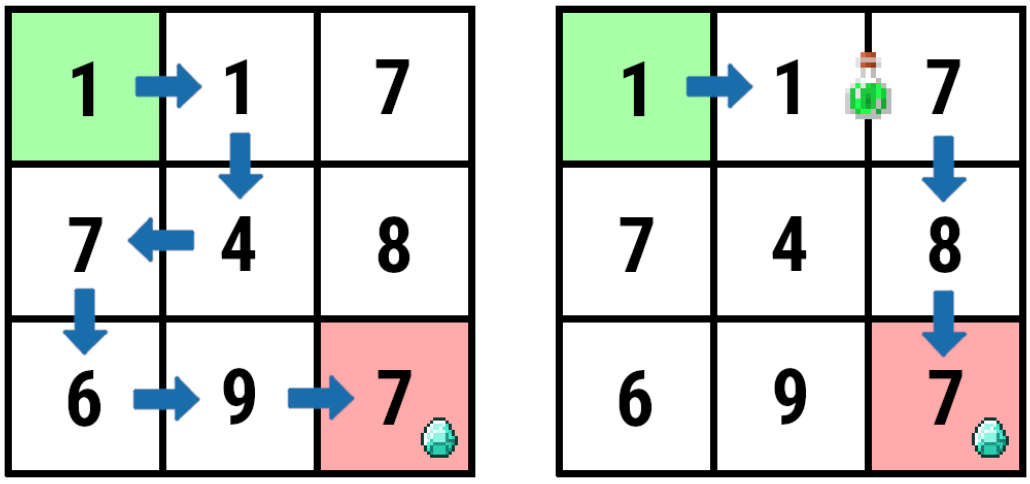

Figure 1: Sample inputs.

Problem E
Eradicative Expressions

Time limit: 2 seconds
Memory limit: 512 megabytes

Problem Description

Shiba Tatsuya lives in a universe where magic exists and has been evolving with the world along with technology. As a magician, Tatsuya is able to cast multiple magic spells with various effects. For instance, "Material Burst" (a.k.a. 質量爆散〈マテリアル・バースト〉) is one of his most famous spells due to its sheer destructive ability. It is never in one's best interest to use it without thorough planning, as a single careless cast can easily blast a planet into oblivion.

There are k characters in the alphabet of magicians, which happens to be a set of lowercase English letters, sorted in their usual order (`a` to `z`). In addition, there is a *dictionary* consisting of *words* that can be used for chanting. Each *word* is a string of length k and is a permutation of all k characters. The *dictionary* is a collection of all $k!$ possible words, sorted in lexicographical order. Recall that the lexicographical order is an ordering of strings such that for any two strings s and t, s appears before t if and only if:

- s is a prefix of t, or
- there is an index i such that $s_i < t_i$ and $\forall\, 1 \leq j < i$, $s_j = t_j$.

In order to cast a magic spell, one has to construct an *activation sequence* by manipulating "psions", which, in short, are particles that can further start processes which overwrite or interfere with physical rules and therefore alter reality. Each *activation sequence* of Tatsuya's "Material Burst" is a permutation of n words $S_1, S_2, \ldots, S_n$, which all belong to the dictionary and are **not** necessarily distinct. For a word w, its potential $f(w)$ is defined as the index of w in the dictionary.

The destructive power of Tatsuya's "Material Burst" depends on the activation sequence he chooses. For each permutation $p_1, p_2, \ldots, p_n$ of integers from 1 to n, the sequence $S_{p_1}, S_{p_2}, \ldots, S_{p_n}$ is a valid activation sequence, and its "power" is given by the value

$$\sum_{i=0}^{n-1} (f(S_{p_{i+1}}) - f(S_{p_i})) \times W_i$$

where $W_1, W_2, \ldots, W_{n-1}$ are constants.

Tatsuya wants to control the power of the "Material Burst" he uses. For an integer x, he wants to know the x^{th} largest value among the powers of all $n!$ activation sequences (Note: there might be duplicate sequences! See the Hint section for an example.). Your task is to find out the answer for multiple x's.

Input Format

The first line contains three integers n, k, q, representing the number of characters, number of words and number of queries you have to answer.

Then n lines follow, the i^{th} of which contains the string S_i.

The next line contains $n-1$ space-separated integers, denoting the constants $W_1, W_2, \ldots, W_{n-1}$.

Then there are q lines: on the i^{th} line there is an integer x_i denoting a query.

Output Format

Print q lines. On the i^{th} line, you should print the answer of the i^{th} query, i.e. the x_i^{th} largest power of all activation sequences.

Technical Specification

- $2 \leq n, k \leq 8$
- $1 \leq q \leq n!$
- $1 \leq x_i \leq n!$
- $-10^9 \leq W_i \leq 10^9$
- Each string S_i consists of k different lowercase English letters.
- The strings $S_1, S_2, \ldots, S_n$ contain the same set of characters.

Sample Input 1

```
2 2 2
ab
ba
100
1
2
```

Sample Output 1

```
100
-100
```

Sample Input 2

```
3 4 4
abdc
dcba
abdc
-1 100
4
6
1
2
```

Sample Output 2

```
22
-2222
2200
2200
```

Hint

In the first sample, the alphabet is $\{\texttt{a}, \texttt{b}\}$ and the dictionary is $[\texttt{ab}, \texttt{ba}]$. The potentials are $f(\texttt{ab}) = 1$ and $f(\texttt{ba}) = 2$ respectively.

- The power of the activation sequence "`ab ba`" is $(2 - 1) \times 100 = 100$.
- The power of the activation sequence "`ba ab`" is $(1 - 2) \times 100 = -100$.

Therefore the 1^{st} largest power is 100 and the 2^{nd} largest power is -100.

In the second sample, the alphabet is $\{\texttt{a}, \texttt{b}, \texttt{c}, \texttt{d}\}$ and the dictionary has 24 words. The potentials are $f(\texttt{abdc}) = 2$, $f(\texttt{dcba}) = 24$ and $f(\texttt{abdc}) = 2$. There are $3! = 6$ possible activation sequences:

Permutation	Sequence	Power
S_1, S_2, S_3	"`abdc dcba abdc`"	-2222
S_1, S_3, S_2	"`abdc abdc dcba`"	2200
S_2, S_1, S_3	"`dcba abdc abdc`"	22
S_2, S_3, S_1	"`dcba abdc abdc`"	22
S_3, S_1, S_2	"`abdc abdc dcba`"	2200
S_3, S_2, S_1	"`abdc dcba abdc`"	-2222

Almost blank page

Problem F
Frontend Failure

Time limit: 2 seconds
Memory limit: 512 megabytes

Problem Description

LYS is a senior backend web developer at BambooFox LLC, a company that provides one of the finest penetration testing services in the world. (Disclaimer: it doesn't actually exist.) His job (backend development) includes server-side scripting, database management, etc.; in general, any logic behind the interface of a website, usually far away on the internet, beyond a user's reach.

Frontend development, on the other hand, is the construction of components that users directly interact with. Usually, this refers to the interface of a website. A crucial part of frontend development is design, as appearance and usability can both greatly affect user experiences and thus determine a website's quality.

Speaking of web design, many developers work with CSS (Cascading Style Sheets), a very popular language used to tweak the layout and style of a web page. Although simple and easy to learn, there are only a few people on earth that managed to achieve true mastery of its mechanisms. Unfortunately, LYS, as a professional backend developer, is not one of them.

For some reasons, LYS has to create the frontend of a new website, consisting of one single page. The page is horizontally divided into n sections numbered from 1 to n, where the i^{th} section has its width set to w_i pixels using CSS (negative values are possible because of transforms, but we don't care about how it works here). A segment of length l is defined as a set of l sections with contiguous indices, and the width of a segment is defined as the total width of all sections belonging to it.

After the website is published, LYS has been receiving complaints regarding the layout of his page. He quickly realized that it was the sections' widths that were causing problems. Every user that visits the page will select some segment of length m and will become unsatisfied if the segment's width is not exactly t, as this is not harmonic. Additionally, if any section has zero width or has width with absolute value greater than 10^9 pixels, the user will also be unsatisfied.

LYS wishes to fix this issue by changing the sections' widths. However, dealing with layout design and CSS doesn't seem like an easy task to him. Can you do it instead?

Input Format

The first line contains an integer T, denoting the number of test cases.

Then T test cases follow, each containing a line with three space-separated integers n, m, t.

Output Format

Output T lines. The i^{th} line should be the answer to the i^{th} case, which consists of n space-separated integers $w_1, w_2, \ldots, w_n$, the widths of sections that can satisfy every possible user, in pixels. If there are multiple answers, you may output any of them.

Technical Specification

- $1 \leq T \leq 100$
- $1 \leq m \leq n \leq 10^4$
- $1 \leq t \leq 10^9$
- $t \geq m$
- This problem has an output limit of 20MB (20,971,520 characters).

Sample Input 1

```
3
10 1 1
10 2 4
10 2 2
```

Sample Output 1

```
1 1 1 1 1 1 1 1 1 1
2 2 2 2 2 2 2 2 2 2
1 1 1 1 1 1 1 1 1 1
```

Hint

This problem is easy!

Let's see the second sample test case:

- $[2, 2, 2, 2, 2, 2, 2, 2, 2, 2]$ and $[1, 3, 1, 3, 1, 3, 1, 3, 1, 3]$ are both valid answers because for each segment of length 2, the sum of its widths is 4.
- $[2, 2, 2, 2, 2, 2, 3, 1, 3, 1]$ is not a valid answer because the widths of the 6^{th} and 7^{th} sections sum to 5 instead of 4.
- $[0, 4, 0, 4, 0, 4, 0, 4, 0, 4]$ is not a valid answer. Although the width of segments are correct, there are sections with zero width so users will be unsatisfied.

Problem G
Geometric Greatness

Time limit: 2 seconds

Memory limit: 512 megabytes

Problem Description

*"Do you want the triangles? Again, triangles." "**Yes.**"*

Hank likes triangles. He even listens to a song to which the lyrics contain the above sentence. Since he loves triangles so much, he received a triangle as his birthday present this year. The triangle Hank received is a right triangle with width a, height b, and hypotenuse $\sqrt{a^2+b^2}$. **In this problem, you only need to consider the case where $a = 1$.**

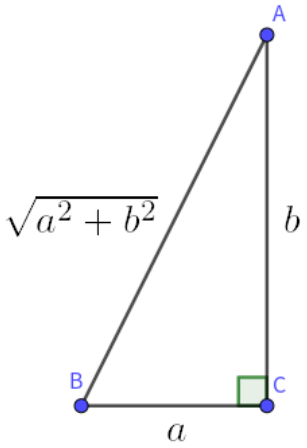

Figure 2: An example of Hank's triangle.

As one triangle is obviously not enough, he wants to construct a sequence of n triangles. The first triangle T_1 of the sequence is the one he received. For $i = 2, 3, \ldots, n$, he creates a new triangle T_i by the following rules:

- T_i's width is the same as T_{i-1}'s perimeter length. The perimeter length of a triangle is the sum of all its side lengths.
- T_i is similar to T_{i-1}. Two triangles X and Y are similar only if their side lengths satisfy
$$\frac{X\text{'s width}}{Y\text{'s width}} = \frac{X\text{'s height}}{Y\text{'s height}} = \frac{X\text{'s hypotenuse}}{Y\text{'s hypotenuse}}.$$

After finishing all n triangles, Hank is curious about the total perimeter length and **twice** the total area of them. Help him write a program to find the answers.

Input Format

The first line contains an integer T, the number of test cases.

Then T lines follow, each representing a test case. On each line, there are three space-separated integers a, b, n.

Output Format

For each test case, output two lines describing the total perimeter length and **twice** the total area of all triangles.

It is guaranteed that each answer can be written in the form $m + k\sqrt{c}$ where m, k are integers and c is a positive integer. If there are multiple possible values for c, choose the representation with minimum positive c. Additionally, if the answer is rational, i.e., $c = 1$, you have to choose $k = 0$.

Print three integers m, k, c on one line for each answer. Since m and k can get really big, output them modulo $10^9 + 7$. You should **not** take the modulo for c.

For example, if the answer is $8+9\sqrt{10}$, you have to print "`8 9 10`" (without quotes). Although $8 + 3\sqrt{90}$ is also correct, you cannot print "`8 3 90`" because the c here is not the minimum positive value for it.

Technical Specification

- $1 \leq T \leq 10^3$
- $a = 1$
- $1 \leq b \leq 10^5$
- $1 \leq n \leq 10^9$

Sample Input 1

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

Sample Output 1

```
3 1 5
2 0 1
89 39 5
782 348 5
35491 8607 17
97145164 23561160 17
323779 78527 17
85494036 961020353 17
6 1 26
5 0 1
```

Hint

In the first test case, there is only one triangle (the one Hank received).

- Its side lengths are 1, 2 and $\sqrt{1^2 + 2^2} = \sqrt{5}$ so the total perimeter length is $3 + \sqrt{5}$. Therefore, you should print "`3 1 5`" on the first line.
- Its area is $\frac{1}{2} \times 1 \times 2 = 1$ so twice the total area is 2. Therefore, you should print "`2 0 1`" on the second line.

Problem H
Hall of Hyper-chairs™

Time limit: 3 seconds
Memory limit: 512 megabytes

Problem Description

Anna has a gathering hall in the outrealm. Time works differently in the outrealm. However, one day in the outrealm is equal to one day on Earth.

Each day in the outrealm is separated into 10^{12} time segments, indexed from 1. Each group may register to use the gathering hall for 1 time segment. No two groups may share the same time segment.

In order to sit in the outrealm gathering hall, you need hyper-chairs. Each person needs 1 hyper-chair, otherwise they don't get to sit at all. Hyper-chairs are affected by the phenomenon called "chair-boom". A chair-boom always happens at the beginning of a day. When a chair-boom happens, it destroys all hyper-chairs in the outrealm. However, after the chair-boom, a portal opens up in the outrealm at the end of some time segment on that day and connects to the chair realm. Anna can transfer infinitely many hyper-chairs for free from the chair realm to the outrealm.

While chair-booms seem unpredictable in the outrealm, they can easily be predicted on the Earth. In order to prevent people from having no hyper-chairs to sit in and leaving bad reviews, Anna has to send hyper-chairs into the outrealm from Earth. She does this by having a portal that allows her to send hyper-chairs from Earth right after the chair-boom. Hyper-chairs are expensive to purchase on the Earth, and Anna only wants to send as few chairs possible without leaving anyone without a seat each time a chair-boom happens.

There will only be a new group registration or one chair-boom each day. Anna usually handles all this all on her own, but as the business grows, she asks you to help her handle the requests and chair-boom forecasts.

Input Format

The input starts with an integer n, meaning there are n days. For each day, there is only one task. A task is either a new group registering their time slot or a chair-boom.

Then n lines follow, each representing a task. Each task starts with an integer q.

- If $q = 1$, this means that this task is a new group registration. Then two integers follow, t and c, meaning that a group of c people want to reserve the time segment t.
- If $q = 2$, this means that a chair-boom will occur. There one integer t follows, meaning that after the chair-boom the portal will open at the end of time segment t on that day.

Output Format

For each task with $q = 2$, output the minimum number of hyper-chairs that Anna has to send into the outrealm to allow everyone to have a seat on that day.

Technical Specification

- $1 \leq n \leq 10^6$
- $q \in \{1, 2\}$
- $1 \leq t \leq 10^{12}$
- $1 \leq c \leq 10^9$

Sample Input 1

```
10
1 1 5
1 3 6
1 10 9
2 4
2 2
2 10
1 5 12
2 10
1 6 7
2 6
```

Sample Output 1

```
6
5
9
12
12
```

Hint

After the first 3 days, we know of the following reservations and how many people will be in the hall at each time segment:

time:	1	2	3	4	5	6	7	8	9	10	11	...
count:	5	0	6	0	0	0	0	0	0	9	0	...

On day 4, a chair-boom happens. For time segments 1 to 4, the maximum people in the hall at once is 6, so Anna need to send in 6 hyper-chairs. For the group at time segment 10, Anna may transfer hyper-chairs from the portal connected to the chair realm.

On day 5, another chair-boom happens. We only need to send 5 hyper-chairs for the group at $t = 1$. Since the chair-boom happens at the beginning of the day and removes all hyper-chairs, we cannot rely on the previously sent hyper-chairs.

On day 6, we now need to send 9 hyper-chairs after the chair-boom, to satisfy the group at $t = 10$. The portal opens at the end of time segment 10, so some people don't have hyper-chairs if we send less than 9.

After day 7, the people that are in the hall at each time segment is updated:

time:	1	2	3	4	5	6	7	8	9	10	11	...
count:	5	0	6	0	12	0	0	0	0	9	0	...

Note that since time works differently, and outrealm gates can lead to different time segments, we can have a later group reserve an earlier time segment.

On day 8, we need to send 12 hyper-chairs after the chair-boom for time segment 5.

After day 9, the people that are in the hall at each time segment is updated:

time:	1	2	3	4	5	6	7	8	9	10	11	...
count:	5	0	6	0	12	7	0	0	0	9	0	...

On day 10, a chair-boom happens. We need to send 12 hyper-chairs for time segment 5.