Problema 1 - Giga-Kilo-Gigabyte

As you probably already know one byte is 8 bits, 1024 bytes is 1 kilobyte, 1024 kilobytes are 1 megabyte, etc.

The most common known prefixes to display units are the following:

|  |  |
| --- | --- |
| 1000 units | 1 kilounit |
| 1000 kilounits | 1 megaunit |
| 1000 megaunits | 1 gigaunit |
| 1000 gigaunits | 1 teraunit |
| 1000 teraunits | 1 petaunit |
| 1000 petaunits | 1 exaunit |
| 1000 exaunits | 1 zettaunit |
| 1000 zettaunits | 1 yottaunit |

Due to the fact that 2^10=1024 is the closest number to 1000, we say that 1024 bytes are 1 kilobyte, for this problem we will ignore this rule of thumb and assume that 1000 bytes are 1 kilobyte.

In the country of *Diegolandia* it is widely known that 1 zettabyte is 10^21 bytes, however Diego knows that it’s possible to prepend different prefixes and to stack them together to represent a higher amount, for example 1 gigabyte can be 1 kilo-kilobyte (1000 kilobytes). In the case of the zettabyte, Diego knows it can be represented as 1 giga-kilo-gigabyte (10^9\*10^3\*10^9=10^21), it can also be represented a 1 kilo-giga-gigabyte, or 1 exa-kilobyte, or 1 kilo-exabyte, etc.

In this problem we only care about how many different ways Diego can prepend the different prefixes (all the way from kilo to yotta), since this number might be huge print it modulo 10^9+7.

**Input**

An integer T the number of test cases (1<=T<=10^5).

Followed by T lines, each with 1 number N (3<=N<10^6), which means that Diego wants to know in how many ways he can prepend the different known prefixes to display units to represent the number 10^N. It is guaranteed that N will be divisible by 3.

**Output**

For each test case print a single line with a single integer representing the amount of ways Diego can prepend the known prefixes to display units to represent the number 10^N, remember to print this number modulo 10^9+7

**Sample input**

5

3

6

9

12

24

**Sample output**

1

2

4

8

128

**Problema 2 -**Three-course meal

You go to a fancy restaurant with exactly K out of your N friends, this restaurant offers a three-course meal, first comes the appetizer, then the main entrance, and lastly dessert. Since this is a fancy restaurant you can’t start the next course until everyone in the table finishes eating their current dish.

You want to choose which friends are going to dinner with you in such a way that everyone in the table finishes the three-course meal as soon as possible. You can assume you will always eat equally fast or faster than the fastest eaters among your friends and that the time to serve and prepare all the dishes is 0.

**Input**

Two integers N and K (1<=N<=2000, K<=N), the amount of friends you have and the exact amount of friends that are going to dine with you.

N lines, each containing the integers, A, E, D (1<=A,E,D<=10^9), the amount of time in minutes that it takes for each of your friends to eat their appetizer, main entrance and dessert, respectively.

**Output**

A single integer, the least amount of time it will take to eat the three course meal with K of your friends.

**Sample input 1**

3 1

1 1 1

1 2 1

1 1 2

**Sample output 1**

3

**Sample input 2**

3 2

1 1 1

1 2 1

1 1 2

**Sample output 2**

4

**Sample input 3**

5 3

5 7 9

3 11 5

9 4 1

1 5 9

2 4 2

**Sample output 3**

21

Problema 3 - Basic Encryption

Diego is trying to send a private message to someone, but this message is going to be publicly sent, because of this he needs a way to encrypt it. Usually he would use a secure algorithm, for example RSA, but this time around he is going to encrypt the message in such a way that it doesn’t matter if the receiver has a computer or not, she can decrypt the message.

You are an internationally renowned hacker and because of that you are going to intercept Diego’s messages and need a way to decrypt them, but doing it by hand is a hassle, so you will write a program that does this for you.

Diego encrypts in the following way, he has an integer S (0<=S<=25), originally S=0. For each alphabetic letter he adds the value of the current letter to S and then prints the letter that matches the value of S. If the value of S ever exceeds 25 Diego reduces the value by 26. The value of each letter is established in the following manner: a is worth 0, b is worth 1, …, z is worth 25. Keep in mind that lower and upper case letters have the same value, and when encrypted they should stay as either capital or lower case letters, depending on what the letter was. Also, any character that is not alphabetic doesn’t get affected by S and stays the same.

**Input**

A single integer S (1<=S<=10^9), followed by several lines (read until EOF), each line contains the encrypted message Diego sent to his contact. Keep in mind that each line will contains alphabetic characters (both upper and lowercase), digits, spaces and punctuation marks. You are guaranteed that the sum of the lengths of all lines doesn’t exceed 10^3.

**Output**

For each case print a single line containing the decrypted string.

**Sample input**

15

Wdap!!

Ta eapc thip hpaxtcsd st prjtgsd p ad eapctpsd.

Tc 24 wdgph edsgtbdh strxg fjt ta eapc ujt jc tmxid.

**Sample output**

Hola!!

El plan esta saliendo de acuerdo a lo planeado.

En 24 horas podremos decir que el plan fue un exito.

Problema 4 - Freddy and minifier

Freddy has a very big elephant, Simon. He is flying to Gabon in order to let Simon meet his brothers and sisters, however he is very big, which means he doesn’t really fit in the plane. In order to make him fit Freddy needs to minify his size. In order to do that he has N minifying guns (they make stuff smaller). However, each gun can only be used once. Each gun has a cost of using it c\*w, where c is unique per gun and w is the weight of the thing they are minifying. Additionally each gun has a minify ratio called r, which means that the size and weight get reduced to r percent of the original value. Given that the initial weight of Simon is W and he is going to get shrunk by each of the N guns exactly once, what is the minimum cost of doing this?

**Input**

Two integers, N and W (1<=N<=10^5, 1<=W<=10^4).

Followed by N lines each with 2 integers c (1<=c<=10^4) and r (0<=r<=1), the c and r values of the ith gun.

**Output**

A real number, the minimum cost of minifying Simon with each of the N guns exactly once. Your answer will be considered correct if its relative or absolute error doesn’t exceed 10^-6

**Sample input 1**

2

7 0.5

9 0.2

**Sample output 1**

10.4

First use the second gun, then the first one.

**Sample input 2**

2

7 0.5

90 0.2

**Sample output 2**

52

First shoot with the first gun, then the second one.

Problema 5 - Perfect polygon

Diego is looking through a window of size W\*H and he wonders if given that particular window if there exists a perfect polygon for it. The answer is obviously yes, because perfection is relative to a specific metric. Diego decides that the polygon will be considered perfect if it’s impossible to increase its width and height without leaving the window.

Diego already has a polygon but he doesn’t really like it that much, so first he will rotate it counterclockwise by some angle A, and then adjust the overall width and height of the polygon in order to make it fit perfectly within the window.

In other words, given some polygon P you have to rotate it A degrees counterclockwise and then scale values in the x axis of the polygon to make sure that the point with the lowest x value is in position 0 and the biggest x is in position W, then do the same for the ys, but now the biggest is in position H.

**Input**

Four integers A, W, H, and N (0<=A<=359, 1<=W,H<=10^9, and 3<=N<=10^5), the angle (in degrees) to rotate the polygon, the width and height of the window, and amount of points the polygon has.

Then N lines with 2 integers each, the x and y (0<=x,y<=10^9) position of the ith point. The polygon may be concave and self-intersecting. You are guaranteed the polygon has area.

**Output**

N pair of numbers (1 pair per line), the x and y position of the ith point in the original polygon after applying the rotation and fitting the polygon to the window. You answer will be considered right if the relative or absolute error doesn’t exceed 10^-6.

**Sample input 1**

0 5 5 4

1 1

0 1

0 0

1 0

**Sample output 1**

5.000000000 5.000000000

0.000000000 5.000000000

0.000000000 0.000000000

5.000000000 0.000000000

**Sample input 2**

90 5 5 4

1 1

0 1

0 0

1 0

**Sample output 2**

0.000000000 5.000000000

0.000000000 0.000000000

5.000000000 0.000000000

5.000000000 5.000000000

**Sample input 3**

180 5 5 4

1 1

0 1

0 0

1 0

**Sample output 3**

0.000000000 0.000000000

5.000000000 0.000000000

5.000000000 5.000000000

0.000000000 5.000000000

Problema 6 - Minimum played times

Diego is looking at some stats from a game and he sees percentages of success for given events. Diego wonders what is the minimum amount of times each event had to happen in order to get that success percent.

**Input**

An integer N (1<=N<=10^5), the number of events in the game.

N real numbers, each in its own line and with maximum 4 decimals after the point. Note: You should consider that numbers provided in the input are not ceiled, rounded nor truncated. (0<=pi<=1)

**Output**

N integers, the minimum amount of times each event had to occur in order to get that percent.

**Sample input**

4

0.20

0.50

0.025

0.3333

**Sample output**

5

2

40

10000

**Explanation**: The first event has 20% success rate which means that there need to be minimum 5 ocurrances (4 fails and 1 success), the second one has 50% so 1 fail and 1 success, the third one needs 1 success and 39 fails and the fourth one needs 10,000 because the input is not rounded, ceiled nor truncated.

Problema 7 - A+B+C

Everybody knows the typical A+B problem, this one is much harder, instead of A+B it’s going to be A+B+C, but with fractions.

**Input**

An integer T (1<=T<=10^3), the number of test cases

T lines follow, each with 3 fraction numbers, each in the form n/d (1<=n,d<=10^6)

**Output**

One line per test case, a single simplified fraction in the format a/b

**Sample input**

2

1/2 1/3 1/4

3/2 1/2 2/2

**Sample output**

13/12

3/1

Problema 8 - Diego and drinks

Today is 4th of july and because of that Diego is celebrating with his friends, they are all drinking their favorite drink in a cup made out of glass. Glass has a property that when you touch it with another thing made out of glass they will produce a sound “clink!”. Everytime that someone opens up a drink they go and touch everybody else's drink, each person drinks at their own pace and as soon as they finish their drink they grab another one and “clink!” everyone in the party before anybody else finishes their drink. You can assume that everyone always has a drink and that in the beginning all of them grabbed drinks then “clink!” each other once. Diego knows 2 things, he has N friends in the party and he heard exactly K “clink!”s

**Input**

T an integer T (1<=T<=10^5), the amount of 4th of july parties that Diego went to.

Followed by T lines, each with 2 integers N (1<=N<=10^4) and K (1<=K<=10^9).

**Output**

1 integer for each test case, the amount of drinks that people drank in the party. If it’s not possible to get that amount of “clink!”s then output “Too drunk to count”

**Sample input**

5

1 1

2 5

2 6

3 15

1 0

**Sample output**

2

4

Too drunk to count

7

Too drunk to count