DAPC 2022

Delft Algorithm Programming Contest 2022
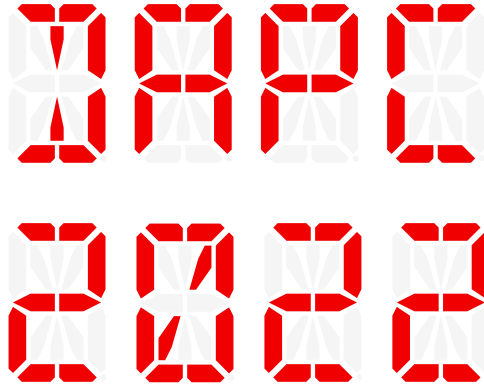
Problems

A   Abbreviated Aliases

Time limit: 2s

You are the owner of a large successful internet site with lots of users. All these users have chosen an alias of exactly l characters for logging into the site. Recently, you noticed that you started running into disk space issues: storing all these aliases takes up a lot of data!

You do not have enough money to buy extra storage, so you are looking for ways to reduce the storage space needed. A friend gives you the following compression idea that might help: instead of storing the full alias for each user, you might get away with only storing a prefix of that alias, as long as no other alias has the same prefix. For example, if you just have the aliases `james` and `jacob`, you can store only `jam` and `jac` and still be able to identify them both.

Register

Could not register alias:
Storage limit reached.

Alias
jacob

Password
....

Register

CC BY-SA 3.0 by Wikimedia, modified

This idea sounds quite interesting to you, and you are looking forward to finally having more space available on your disk again. You would like to find out how much space you need to store all aliases using this compression technique.

Input

The input consists of:

- One line with two integers n and l ($2 \leq n \leq 10^4$, $1 \leq l \leq 10^3$), the number of aliases and the length of each alias.

- n lines, each with an alias: a string consisting of exactly l English lowercase characters (`a-z`). Each alias is unique.

Output

Output the total number of characters you still need to store if you apply this compression technique.

Sample Input 1	Sample Output 1
2 5 james jacob	6

Sample Input 2	Sample Output 2
4 4 xxxx yxxx xxyx yxxy	14

This page is intentionally left blank.

B   Bubble-bubble Sort

Time limit: 2s

Bubbles! As a fanatical supporter of the Bubbles Are Perfect Creatures movement, you have accumulated a large collection of bubbles in all colours and sizes. Being a long time member, your bubbles are among the best in the world, and now is the time to show this. Tomorrow, the yearly Bubble Exposition will be held, and your goal is to win the Bubble Prize and become the Bubble Champion!
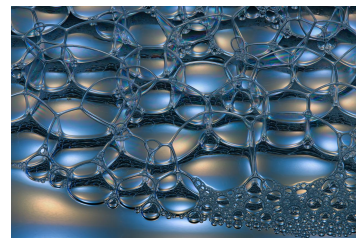
Image by Tom from Pixabay.

However, this is not an easy competition. In order to win, you do not only need the most beautiful bubbles, you also need the best-looking placement of bubbles. You have decided to order the bubbles by bubbliness: less bubblier bubbles to the left, more bubblier bubbles to the right. However, it is hard to compare all the bubbles in your collection at once. In fact, you can only compare up to k bubbles by eye before losing track of all the bubbles. Since your collection consists of more than k bubbles, you need a fancier sorting algorithm.

Your first thought is to use the best sorting algorithm for bubbly purposes, namely Bubble Sort. However, this is the most prestigious bubble competition, so you decide to do better: Bubble-bubble Sort. It works as follows.

Initially, your bubbles are placed in an arbitrary order. Every hour, you do the following: you look at the first k bubbles and place them in the optimal order. Then, you look at bubbles 2 to $k + 1$ and place those in the correct order. Then, you look at bubbles 3 to $k + 2$, and so on, until you have placed the last k bubbles in the correct order. You then admire how the bubble collection looks so far until the next hour begins and you start at the first bubbles again.

Is this algorithm fast enough to place all your bubbles, or do you need to go further and invent a Bubble-bubble-bubble Sort algorithm? To be precise, after how many hours are the bubbles in the optimal positions?

Input

The input consists of:

- One line with two integers n and k ($2 \leq k < n \leq 2500$), the number of bubbles and the number of bubbles you can sort at once.

- One line with n integers a ($0 \leq a \leq 10^9$), the bubbliness of each bubble in the initial placement of your bubble collection.

Output

Output the number of hours needed to sort your bubble collection.

Sample Input 1

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

Sample Output 1

```
3
```

Sample Input 2

```
8 3
60 8 27 7 68 41 53 44
```

Sample Output 2

```
2
```

Sample Input 3

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

Sample Output 3

```
3
```

C   Cookbook Composition

Time limit: 2s

The world-famous chef Gordon Oliver is composing a new cookbook called "Becoming A Perfect Chef". He has a list of recipes that he wants to publish in the cookbook. Each recipe is in the form of a list of steps, where every step might depend on some previous steps (meaning a step cannot be started until all its dependencies have finished), and expected time per step.

Gordon knows that, as an expert chef, he can multitask and do as many tasks simultaneously as needed. Meanwhile, a beginner can do one task at a time, so they need to execute

Chef Gordon Oliver in his natural habitat, multitasking his heart out. CC BY 2.0 by Jeremy Noble on Flickr

them sequentially. He would like to order the recipes for the cookbook by accessibility, where the lowest $\frac{\text{beginner time}}{\text{expert time}}$ ratio recipes come first.

As an example, consider the first sample case. For the oven dish, an expert chef like Gordon Oliver can prepare the tomatoes, eggplants, and sauce all at the same time (with the sauce taking the longest: 5 time), and followed by arranging (1) and baking (30) the dish, this takes $5 + 1 + 30 = 36$ time. On the other hand, a beginner needs $2 + 2 + 5 + 1 + 30 = 40$ time to make the oven dish. This makes the accessibility ratio of the oven dish $40/36 \approx 1.11$. The accessibility ratio of the ice cream is 1 (because beginner and expert chefs both require $5 + 5 + 5 + 240 = 255$ time to prepare it), so it comes before the oven dish in the cookbook.

Input

The input consists of:

- One line with an integer n ($2 \leq n \leq 500$), the number of recipes.

- Then, for every recipe:
  - One line with the name of the recipe and an integer s ($1 \leq s \leq 50$), the number of steps in the recipe.
  - s lines, one for every step in the recipe, with the step name, an integer t ($1 \leq t \leq 10^6$), the step duration, an integer d ($0 \leq d \leq 49$), the number of dependencies, followed by a list of step names that this step depends on.
    A step only appears once all the steps that it depends on have been listed.

The recipe and step names consist of at most 10 English lowercase letters (a–z).

The recipe names are unique and the step names are unique per recipe.

Output

Output the names of the recipes in the cookbook, ordered by accessibility.

If there are multiple valid solutions, you may output any one of them.

Sample Input 1

```
2
ovendish 5
tomatoes 2 0
eggplants 2 0
sauce 5 0
arrange 1 3 tomatoes eggplants sauce
bake 30 1 arrange
icecream 4
mix 5 0
heat 5 1 mix
churn 5 1 heat
freeze 240 1 churn
```

Sample Output 1

```
icecream
ovendish
```

Sample Input 2

```
2
recipea 4
stepa 5 0
stepb 5 1 stepa
stepc 2 0
stepd 2 1 stepc
recipeb 4
stepa 1 0
stepb 2 1 stepa
stepc 2 1 stepa
stepd 1 2 stepb stepc
```

Sample Output 2

```
recipea
recipeb
```

Sample Input 3

```
2
recipea 2
stepa 2 0
stepb 2 1 stepa
recipeb 2
stepa 5 0
stepb 5 1 stepa
```

Sample Output 3

```
recipeb
recipea
```

D   Dimensional Debugging

Time limit: 2s

After struggling with this *one* problem for *days*, you have had enough! You are determined to find the bug in your algorithm once and for all! To do so, you will start all over. From scratch. At least you are sure you know the correct answer in the most trivial case: the answer in $(0, 0, \ldots, 0)$ is 0.

You will re-solve the problem, which takes k parameters, using n simpler but slower algorithms. Each algorithm has two bounds for every parameter i (L_i and H_i). An algorithm is only fast enough to run on inputs $(x_1, \ldots, x_k)$ where $x_i \leq H_i$ for all parameters i. You are confident the implementation of an algorithm is correct if you can verify its correctness at least once on an input $(x_1, \ldots, x_k)$ where $x_i \geq L_i$ for all parameters i. To do so, you will need another algorithm that you already proved to be correct and can handle such large inputs, or your knowledge of the answer for $(0, 0, \ldots, 0)$.

Given a list of algorithms and their bounds, find the number of algorithms you are sure are correctly implemented.

As an example, consider the first sample case shown in Figure D.1 on the left. The first algorithm (red, bottom left) can be used to verify the correctness of the second (yellow, top left) and third (blue, bottom right) algorithms. No algorithm can be used to verify the correctness of the fourth algorithm (grey, top right).
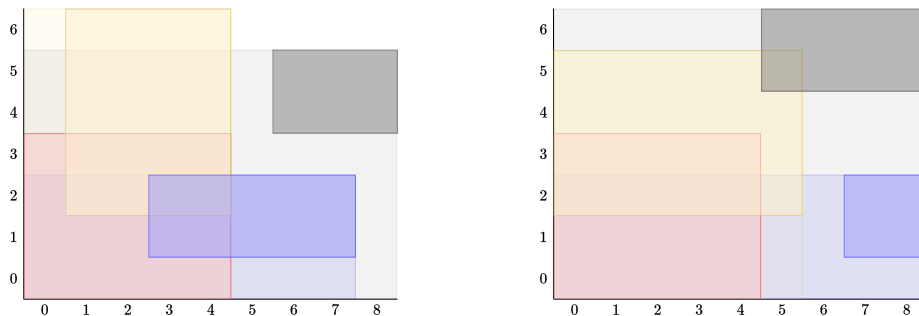

Figure D.1: The algorithms to be tested in samples 1 and 2, respectively. The boxes indicate the parameters where an algorithm must be tested, while the lighter background indicates the region where an algorithm can be used to verify other algorithms.

Input

The input consists of:

- One line with two integers n and k ($1 \leq n \leq 1000$, $1 \leq k \leq 10$), the number of algorithms to test and the number of parameters.

- Then follow n pairs of lines:

   - One line with k integers $L_i, \ldots, L_k$ ($0 \leq L_i \leq 10^9$ for all i).
   - One line with k integers $H_1, \ldots, H_k$ ($0 \leq H_i \leq 10^9$ for all i).

   It is guaranteed that $L_i \leq H_i$ for all $1 \leq i \leq k$.

Output

Output the number of algorithms of which you can verify the correctness.

Sample Input 1	Sample Output 1
4 2 0 0 4 3 1 2 4 6 3 1 7 2 6 4 8 5	3

Sample Input 2	Sample Output 2
4 2 0 0 4 3 0 2 5 5 7 1 8 2 5 5 8 6	4

Sample Input 3	Sample Output 3
3 1 1 10 10 100 0 1	3

Sample Input 4	Sample Output 4
3 3 0 0 1 2 2 1 1 0 0 2 3 4 0 1 0 3 4 5	0

E   Extended Braille

The Blind Association for Pretty Calligraphy is annoyed by the lack of emoticons and math symbols in the braille alphabet. Given that the braille alphabet is supported by the Unicode format, it only makes sense to make all Unicode characters supported in braille.

The goal is to extend the braille alphabet to include all Unicode characters. Of course, this will not fit in the standard 2×3 format, so using a bigger box is allowed. Important is that no two braille characters are the same up to translation, i.e., have the same shape. See Figure E.1 for an example. You let a designer make up a large braille alphabet, and your job is to check how many unique shapes there are among the characters.
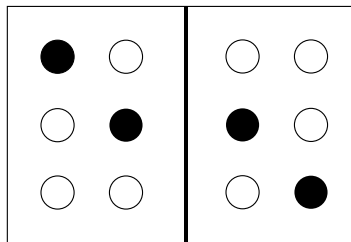

Figure E.1: Illustration of Sample Input 1:
two characters with the same shape.

Input

The input consists of:

- One line with an integer n ($1 \le n \le 10^5$), the number of braille characters.

- Then for each of the n braille characters:

  - One line with an integer m ($1 \le m \le 1000$), the number of dots.

  - m lines, each with two integers x and y ($|x|, |y| \le 1000$), the coordinates of the dots.

The total number of dots is at most 10^6.

Output

Output the number of distinct braille characters up to translation.

Sample Input 1

```
2
2
0 2
1 1
2
0 1
1 0
```

Sample Output 1

```
1
```

Sample Input 2

```
2
3
-1 0
0 1
1 0
3
-1 0
0 -1
1 0
```

Sample Output 2

```
2
```

F Fastestest Function

Time limit: 1s

You are working as a software developer for the Bug Acquisition Programming Company. They developed a specific piece of software called Program C that they sell to their clients. For the past weeks, you have been working on optimising a specific function `foo` in the main code path in Program C. You have made it a lot faster and would like to show off to your boss about it.
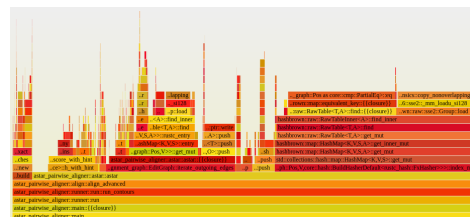
A flamegraph.

Your IDE has a nice tool that allows you to profile your code and tell you what percentage of the total running time `foo` takes. You can run this on the version before your change and after your change. However, you think it looks a lot cooler if you can just tell your boss how much faster you have made `foo` itself.

Input

The input consists of:

- One line with two integers x and y $(0 < x, y < 100)$, where x is the percentage of the total running time that `foo` took before optimising and y the percentage of the total running time it took after optimising.

Output

Output the factor of how much faster `foo` got after your optimization.

Your answer should have an absolute or relative error of at most 10^{-6}.

Sample Input 1	Sample Output 1
75 50	3.0

Sample Input 2	Sample Output 2
50 75	0.3333333333333333

| Sample Input 3	Sample Output 3
50 50	1.0

This page is intentionally left blank.

G Guessing Primes

Time limit: 10s

Your friends are all on board with the hype of a popular game, where you need to guess a five-letter word in six tries. Unfortunately, your language skills are not the greatest, so this game is not really your cup of tea. However, your mathematics skills greatly outmatch your friends', so you start playing a game called "Brave Alternative Primes Challenge"[1] instead. In order to show off your skills to your friends, you decide to write a program that will always beat the game.

In this game, you need to guess a secret prime number of five digits (i.e., between 10^4 and 10^5) in six turns. After guessing a prime number, you will receive a response consisting of five characters, each corresponding to a single digit in your guess:
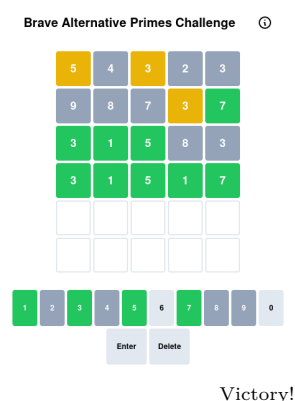
- "g" (green) means you guessed the corresponding digit correctly;

- "y" (yellow) means that the digit is present in a position that is not yet green, but not at this position;

- "w" (white) means that this digit is neither green nor yellow.

Note that the interactor colours at most one of your guessed digits per digit in the secret answer. If your guess includes more occurrences of a digit than the answer, only some of them will be green or yellow.

You win the game when the response is green for all five digits.

Interaction

This is an interactive problem. Your submission will be run against an *interactor*, which reads from the standard output of your submission and writes to the standard input of your submission. This interaction needs to follow a specific protocol:

The interactor first sends one line with an integer n ($1 \le n \le 1000$), the number of rounds.

Then, for each of the n rounds, your program should make at most 6 guesses, each guess being a prime number of five digits (i.e., between 10^4 and 10^5). The interactor will respond with a string of characters in "wyg", as described above. A round ends when the response is "ggggg".

The interactor is not adaptive, i.e., the secret prime number is fixed during a round.

Make sure you flush the buffer after each write.

A testing tool is provided to help you develop your solution.

Using more than 6 queries in one round will result in a wrong answer.

---

[1]Original at `https://converged.yt/primel/`.

Read	Sample Interaction 1	Write
2		
	54323	
ywyww		
	98737	
wwwyg		
	31583	
gggww		
	31517	
ggggg		
	99991	
wwwwy		
	44449	
wgwgw		
	14143	
ggggg		

H   Heavy Hauling

Time limit: 3s

The warehouse of the Boxes And Parcels Center (BAPC) just received an official warning from the inspector: apparently, it does not conform to the latest safety requirements. In the past, it was allowed to stack multiple boxes at the same shelf location, but due to the potential fire hazard, this is no longer allowed. In a hurry, all employees of the BAPC are roused to move the boxes to distinct positions.

The Parcel Retriever Robot.
CC BY 2.0 by Enrique Dans on Flickr

After moving the boxes, the automated parcel retriever robot needs to be reprogrammed such that it knows the correct location of the boxes. Per box that is moved d positions, it takes d^2 time to do this reprogramming. Of course, the BAPC should be up and running as soon as possible after moving the boxes, so the boxes should be moved in such a way that this total reprogramming time is as small as possible. Calculate the minimal time for the reprogramming for an optimal moving of boxes.

The warehouse of the BAPC is unbounded in both directions.

As an example, consider Figure H.1, corresponding to the first sample case. One box at position -1 is moved to the left, which costs 1 time for the reprogramming. The box at position 4 is moved one position to the right, to make place for one of the boxes at position 3, costing 1 time as well. Two boxes at position 3 are moved to the left (costing 1 and 4), and one box at position 3 is moved to the right (costing 1), making the total cost $1+1+1+4+1=8$.
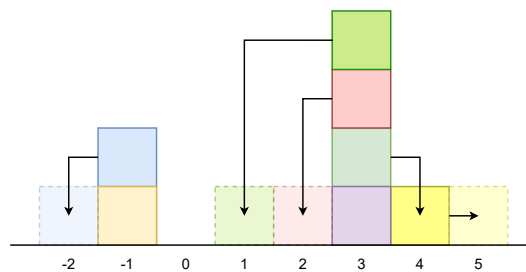

Figure H.1: Visualisation of the first sample case.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 10^6$), the number of boxes.

- One line with n integers x ($|x| \leq 10^9$), the position of each box. The box positions are ordered non-decreasingly.

Output

Output the minimal time to reprogram the parcel retriever robot for an optimal moving of boxes.

Sample Input 1	Sample Output 1
7 -1 -1 3 3 3 3 4	8

Sample Input 2	Sample Output 2
8 2 2 2 2 2 2 4 4	24

I   Inked Inscriptions

Time limit: 4s

The year is 1337. You are a hardworking monk in your abbey, and today you have been tasked to make a copy of your abbey's psalm book. However, there is a problem. The old psalm book sorted psalms by *age*: each time a new psalm made its way into the abbey, it was added in the back. The head monk wants the new book to be sorted by *title* instead to make specific psalms easier to find. This means that you need to write all psalms on different pages in the new book! Since there can be a lot of psalms (each fitting on one page), this requires some careful planning.
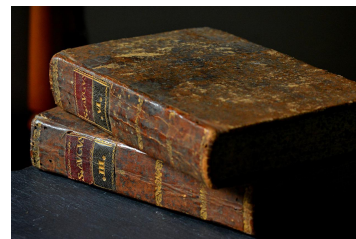
Image by José David Castillo Arias from Pixabay.

To copy a psalm, both books need to be opened on the proper page.[2] For example, suppose that you want to copy a psalm from page 5 of the old book to page 8 of the new book. Also suppose that the old book is currently opened on page 12 and the new book is opened on page 3. Then, you need to flip $|12 - 5| = 7$ pages in the old book and $|3 - 8| = 5$ pages of the new book to arrive at the proper pages. This takes $7 + 5 = 12$ page flips in total. Since books are very fragile and valuable, you want to limit the number of page flips needed to copy all the psalms.

Both books are initially opened on page 1. In which order should you copy the psalms to ensure that you use at most $2n\sqrt{n}$ page flips (rounded up)?

Input

The input consists of:

- One line with an integer n ($1 \le n \le 10^4$), the number of psalms.

- One line with a permutation of the integers 1 to n. If the i-th integer is j, then you should copy the psalm on page i of the old book to page j of the new book.

Output

Output n pairs of integers, where each pair i and j indicates that the psalm on page i of the old book should be copied to page j of the new book.

Each psalm should be copied exactly once and onto the correct page. The total number of page flips needed to perform these instructions should be at most $2n\sqrt{n}$, rounded up.

If there are multiple valid solutions, you may output any one of them. The number of required page flips does not need to be minimal.

---

[2]Note that psalms are only written on the right-hand pages, and therefore, only the right-hand pages are numbered.

Sample Input 1

```
3
2  1  3
```

Sample Output 1

```
2  1
1  2
3  3
```

Sample Input 2

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

Sample Output 2

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

J   Jabbing Jets

Time limit: 1s

You have just gotten a new job at the Bathroom Accessories Production Company. The first task you are given is to jab holes into showerheads. To prove yourself, you have decided you want to create as many holes as possible.

However, you cannot just randomly drill holes everywhere in the shower-head.[3] In order to ensure that the showerheads look aesthetically pleasing, the company has composed some guidelines which you will have to follow. See Figure J.1 for some examples of aesthetically pleasing showerheads.

- The holes should be arranged in concentric circles of radii $r_1, r_2, \ldots, r_n$: the center of every hole should be on one of these circles.

- The distance between the centers of any two holes should be at least e.
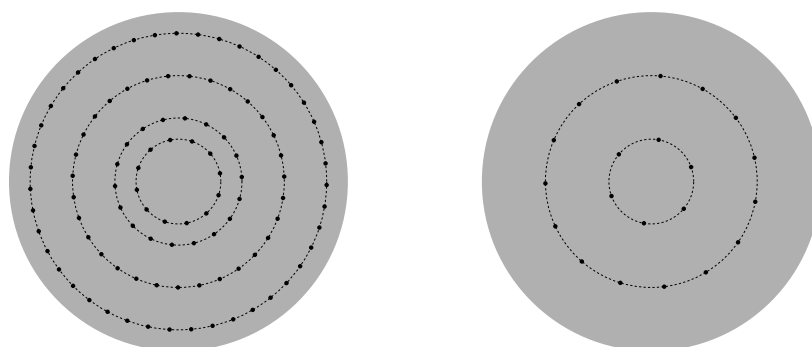
How many holes can you make at most?

Figure J.1:   Possible aesthetically pleasing showerheads for the first two samples.

Input

The input consists of:

- One line with two integers n and e ($1 \leq n, e \leq 10^4$), the number of circles and the minimal distance between holes.

- One line with n integers $r_1, \ldots, r_n$ ($1 \leq r_i \leq 10^4$), the radii of the circles.

It is guaranteed that the numbers r_i are given in increasing order, and that $r_{i+1} - r_i \geq e$. Furthermore, it is guaranteed that increasing any radius r_i by at most 10^{-6} will not change the final answer.

---

[3]At least, not without getting fired.

Output

Output the maximal number of holes that you can make in the showerhead.

Sample Input 1	Sample Output 1
4 1 2 3 5 7	104

Sample Input 2	Sample Output 2
2 2 2 5	21

Sample Input 3	Sample Output 3
3 20 14 53 80	44

K   Knitting Patterns

Time limit: 3s

It is the most relaxing hobby of anyone's grandma: knitting! Your grandma uses a lot of wool, even for the most simple patterns that she is knitting. You are sure that she can be more efficient with her wool, so you decide to write a program that calculates the most efficient use of wool.

A knitting pattern is a long list of stitches, where each stitch can use a different colour.[4] There is a cost for starting or ending the use of a certain colour, for using the wool in a stitch, and for letting it strand through the back unused. For a given knitting pattern, calculate the least possible amount of wool required for every colour of wool.

Knitting Grandma.
Pixabay License

As an example, consider the first sample case. There are three colours of wool (red, green, and blue). The red wool is used at the start and at the end of the pattern: it is most efficient to start and stop using the red wool for both parts separately ($4 \cdot 4 = 16$ cost), and it is used in ten stitches ($10 \cdot 2 = 20$ cost), giving a total cost of 36. The green and blue wool have the same cost: start once (4 cost), use for five stitches ($5 \cdot 2 = 10$ cost), strand along the back for four stitches ($4 \cdot 1 = 4$ cost), and stop using the colour (4 cost), giving a total cost of 22.

Input

The input consists of:

- One line with three integers a, b, and c ($1 \leq a < b < c \leq 1000$), the cost of letting the wool strand through the back unused, the cost of using the wool in a stitch, and the cost of starting or ending the use of a colour of wool.

- One line with a string w ($1 \leq |w| \leq 26$), representing the unique letters used for denoting stitch colours.

- One line with a string p ($1 \leq |p| \leq 10^6$), representing the stitch colours of the knitting pattern.

All stitch colours use English lowercase letters (a–z).

Output

Output, for every colour of wool, in the order as they are in w, the amount of wool used in this pattern.

---

[4]Most real-life knitting patterns are two-dimensional, but since you zig-zag through such a pattern while knitting, the input is one-dimensional for simplicity.

Sample Input 1

Sample Input 1	Sample Output 1

`1 2 4`
`rgb`
`rrrrgbgbgbgbgbrrrrr`
	`36`
	`22`
	`22`

Sample Input 2

Sample Input 2	Sample Output 2

`2 4 1000`
`ab`
`abbbbbbbba`
	`2024`
	`2032`

L   Lots of Liquid

Time limit: 1s

You work at a warehouse that sells chemical products, where somebody just placed an order for all the Boron Acetate Phosphoric Carbonate (BAPC) that you have in store. This liquid is stored in many separate lots, in cube-shaped containers, but your client requires the order to be delivered in a single cube-shaped container that fits all the BAPC liquid perfectly. What should be the size of this container?

Some of the cube-shaped containers.
Used with permission from
BeautifulChemistry.net

Input

The input consists of:

- One line with an integer n ($1 \le n \le 10^5$), the number of cube-shaped containers that you have in store.

- One line with n floating-point numbers c ($1 \le c \le 10^9$), the length of one of the sides for each of these containers.

Output

Output the length of one of the sides of the cube-shaped container that will contain all the BAPC liquid.

Your answer should have an absolute or relative error of at most 10^{-6}.

Sample Input 1	Sample Output 1
3 21 28 35	42

Sample Input 2	Sample Output 2
3 22.10 2022 1337	2200.6131345362505

Sample Input 3
3 1.41421356 2.718281828 3.1415926535

Sample Output 3
3.777901284526486