icpc International Collegiate Programming Contest

The 2024 ICPC Southern California Regional Contest

Problem Set

JETBRAINS

icpc global sponsor
programming tools

upsilon pi epsilon
honor society

2024/2025 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 1
Sumset

A *sumset* S is a set of distinct integers created from a different initial set, I, of distinct integers, such that the integers in S are created by adding each pair of unique integers from I and keeping the distinct resulting values.

As an example, a set of 100 distinct integers has 4950 pairs, so one might expect the sumset to be that size. However, if the set has some structure—that is, the numbers are related in some way to each other—many pairs will sum to the same number. The resulting sumset would be smaller in this case.

In the specific case of the initial set $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$, there are 45 pairs, but only 17 distinct integers in the sumset. Pairs $(1, 6)$, $(2, 5)$, and $(3, 4)$ all add up to 7.

Your team is to write a program that determines the size of the sumset for a given initial set. Input to your program starts with an initial value c, $2 \leq c \leq 500$, giving the size of the initial set. c distinct values follow, each an integer n ($0 < n < 10{,}000$). Values are separated by whitespace.

Your program is to print a line with two values separated by whitespace: the number of pairs in the input set followed by the size of the sumset.

Sample Input

```
10
8 5 10
4 9 3 1
2 7
6
```

Output for the Sample Input

```
45 17
```

2024/2025 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 2
Rainbow Bowl Ranges

You have a set of n bowls, arranged in a circle.

You have many balls of various colors. There are m different colors, and you have c_i balls of the ith color.

You want to distribute all the balls into the bowls. To do this, for each color, you choose a contiguous range of bowls of size c_i and place one ball of that color in each bowl in the range. A contiguous range of bowls is a set of consecutive bowls around the circle. Ranges from different colors are allowed to overlap.

A bowl is *rainbow* if it contains one ball of each color. A *rainbow bowl range* is a contiguous range of rainbow bowls that cannot be extended by including another rainbow bowl.

You want to arrange balls in bowls to maximize the number of rainbow ranges. Given the number of bowls and the number of balls of each color, what is the maximum number of rainbow bowl ranges that can be formed?

The first line of input contains two integers, n ($2 \leq n \leq 10^9$), and m ($1 \leq m \leq 10^5$). The next m lines each contain a single integer, c_i ($1 \leq c_i \leq n$).

Print a single integer, the maximum number of rainbow bowl ranges that can be formed.

Sample Input 1

```
4 2
3
3
```

Output for Sample Input 1

```
2
```

Sample Input 2

```
10 11
3
1
4
1
5
9
2
6
5
3
5
```

Problem 2
Rainbow Bowl Ranges (continued)

Output for Sample Input 2

```
1
```

2024/2025 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 3
Auto-Coin-o-Matic

It's finally here! The day you unveil your new invention, the Auto-Coin-o-Matic! You watch with glee and anxiety as people insert their card into the machine, type in the amount they want, and get exact change out with the fewest number of coins.

But was it actually the fewest number of coins? That's how it was programmed, but what if you had a bug? It's okay, you're watching. You decide to randomly pick some transactions and double check that what the machine gave out is indeed the fewest number of coins possible. But, oh no, the machine is running out of certain types of coins! Will it still work correctly?

The input starts with two integers n and m ($1 \leq n \leq 2000$, $1 \leq m \leq 10^5$).

The next line contains n integers, $d_1, d_2, \ldots, d_n$ ($1 \leq d_i \leq 10^5$) representing the denominations of coins available initially. It is guaranteed that all denominations are unique.

The next m lines contain a character c ($c \in \{\texttt{Q}, \texttt{X}\}$) and an integer v ($1 \leq v \leq 10^5$), where c is the type of event and v is the value of the event.

- If c is the character `Q`, this is a query and the output should be the minimum number of coins needed to give out exactly v. It is guaranteed that there will be at least one query. If it is not possible to make v exactly with the available denominations, output -1 instead.
- If c is the character `X`, this means the machine is out of coins of denomination v. All queries after this point cannot use this denomination. It is guaranteed that each `X` event corresponds to a denomination v which the machine currently has in stock.

Output k lines, where k is the number of query events ($c = \texttt{Q}$). On the i^{th} line, output the fewest number of coins needed to give change for the i^{th} query, or -1 if this is impossible.

Sample Input

```
4 7
1 2 5 10
Q 23
X 1
Q 23
X 5
Q 23
X 10
Q 22
```

Output for the Sample Input

```
4
6
-1
11
```

2024/2025 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 4
Unit Conversion

Your younger brother is learning about unit conversions at school. Each day, he comes back with either new knowledge or a homework assignment.

When he gains new knowledge, he learns a conversion rate between two units. The conversion rate will be given by a linear function. Examples include:

- degrees Fahrenheit = 1.8 × degrees Celsius + 32
- Kelvins = degrees Celsius + 273.15
- B = 1024 × kB

When he gets homework, he has to answer unit conversion questions. For example, "How many Kelvins are equal to 36.6 degrees Fahrenheit?" Note that even if he has no direct rule for converting the units, he still may be able to figure out the solution, as in the given example. Sometimes the homework is tricky and the answer cannot be deduced from his current knowledge—he should then say the question is too hard.

As his older brother, you have to help him by checking the correctness of his solutions. That means for every homework question, you have to prepare an answer so he can validate the correctness of his result. That is your task: given the knowledge he learns and the homework he has to solve, output the answers to all the homework questions.

The input consists of at most 2×10^5 lines of knowledge and homework questions. For a new piece of knowledge, the input line contains the following:

`K` *name1* = *a* *name2* $\{+/-o\}$

This specifies a piece of new knowledge: unit *name1* is equal to $a \times$ *name2* $\pm o$. The braces indicate that the part "$+/-o$" can be omitted.

When a homework question is given, the input line contains the following:

`H` *x* *name1* = ? *name2*

The homework question asks how much of the unit *name2* is equal to x of unit *name1*. The names *name1* and *name2* are not necessarily distinct from each other.

Input ends with a line containing a single character "`G`" (for "Graduated!"—a reward for your help with the homework).

It is guaranteed that:

- Each unit name consists of between 1 and 20 upper-case and lower-case letters, digits, and underscores. Case is significant.
- Each value (a, o or x) is a floating point number in the interval [0, 1000], given with three decimal places. Additionally, a will never be equal to 0.
- A new piece of knowledge will never contain a correlation between two units we can already convert based on current knowledge.

At no point in time will there exist a unit that can be converted to over 10^5 in absolute value of a different unit (so we will never know that 1 metric ton can be converted to 10^6 grams).

For each line describing homework, output a line with a single floating-point number, which would make the equation true after replacing the question mark. Your answer will be considered correct if the relative or absolute error does not exceed 10^{-4}. If the answer cannot be provided with the knowledge acquired up to that point, output a line containing only the string "`Too hard!`".

Problem 4
Unit Conversion (continued)

Sample Input

```
K F = 1.800 C + 32.000
K K = 1.000 C + 273.150
H 0.000 C = ? F
H 36.600 F = ? K
K kB = 0.100 kB_by_10
K B = 102.400 kB_by_10
H 1.000 kB = ? b
K b = 8.000 B
H 1.000 kB = ? b
K kilogram = 1000.000 ton
H 1.000 kilogram = ? ton
H 1.000 kg = ? ton
H 1.000 F = ? ton
K 2kg = 2.000 _one_kilogram + 0.000
K pounds_per_inch2 = 14.504 bar
G
```

Output for the Sample Input

```
32.0
275.7055555556
Too hard!
8192.000
0.001
Too hard!
Too hard!
```

2024/2025 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 5
Magic Door

The Arcane Collegium of Mystics has an underground chamber where powerful ancient magic scrolls are kept. While everyone knows of this legendary chamber, only a few high-ranking wizards know its actual location. However, you, a second-year apprentice of the school, accidentally discovered the entrance to the chamber after silently following Master Aldric, the head of the Collegium.

You knew you were not supposed to be there, but curiosity got the better of you, so you decided to hide and watch. Master Aldric stood before the door with n disks embedded in it. Observing carefully, you saw p runic symbols inscribed around each disk. The p symbols were all different, but all disks had the same sequence of symbols, except that the orientations of the disks were not the same.

"Aetheris Vox!"

Master Aldric cast a spell, and the disks on the door began to hum and turn. The disks rotated at different speeds, and some stopped before the others. He kept chanting a series of spells which you never learned. After dozens of spells, the disks began to glow. You noticed that the runic symbols were now perfectly aligned across all the disks. The door slowly opened, and as Master Aldric stepped inside, it quickly shut itself, and the disks turned around randomly. The door was locked once again.

That night, you sneaked outside the living quarters and stood before the entrance to the chamber once more. After casting some spells you knew, you realized that each spell turned the disks counter-clockwise in different amounts. For example, the fire spell turned the first disk by 4 symbols, and the third disk by 6. The second disk did not turn at all. The lightning spell turned only the second disk by 7 symbols.

You also realized that casting the same spell turned the disks in the same way as before. By casting the lightning spell twice more, the second disk rotated by an additional 14 symbols.

After studying the effect of each spell, you wondered whether your limited knowledge of spells could still get you into the chamber. It looked as if the door would open as long as the runic symbols of all disks were aligned, and it did not matter which of the p symbols was at the top at the end.

"Machina Intellegentia!"

You summoned a computer and started writing a program to solve this problem.

The input begins with two space-separated integers n and p, where $1 \leq n \leq 300$ is the number of disks on the door, and $2 \leq p < 10^9$ is the number of symbols on each disk. It is guaranteed that p is a prime number. The symbols are numbered from 1 to p in the order they appear on the disks around the circumference, in clockwise order.

The next line contains n space-separated integers, denoting the initial orientation of the disks. The ith number $1 \leq a_i \leq p$ denotes which of the p symbols is at the top of the ith disk.

The next line contains a single integer $1 \leq m \leq 300$, which is the number of spells you know. Each of the next m lines describes the effect of the spells. A line contains n space-separated integers, which denote for each disk, how many symbols will rotate as a result of casting the spell. Each rotation amount r is in the range $0 \leq r \leq 2 \times 10^9$.

If it is possible to use some sequence of spells to turn the disks so that they are all aligned, print "`Possible`" on a single line. Otherwise, print "`Impossible`" instead.

Problem 5
Magic Door (continued)

Sample Input 1

```
3 5
3 4 1
2
4 0 6
0 7 0
```

Output for Sample Input 1

```
Possible
```

Explanation for Sample Input 1

By casting the first spell once, disks 1 and 3 will turn so that symbol 2 faces up. Then, casting the second spell 4 times will turn disk 2 so it also shows symbol 2 at the top, aligning all the disks.

Sample Input 2

```
3 5
2 3 5
2
4 0 6
0 10 0
```

Output for Sample Input 2

```
Impossible
```

2024/2025 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 6
Word Game

On the Word Game show, Ashley has selected n words and asks Brandon to combine them. Two words s and t can be combined if s has a suffix of length $k > 0$ that is a prefix of t. The result of combining them is a new word made of s concatenated with the last $t - k$ letters of t. If there are multiple values of k that are valid, any can be chosen.

Brandon must repeatedly take a pair of words from the list of words that can be combined, and replace them in the list with the combined word, until the list contains only a single word, and that word is as short as possible. If multiple final words of the same length are possible, Brandon must find the lexicographically first one.

The first line of the input contains a single integer n ($1 \leq n \leq 5$), the number of words to start out with. The next n lines each contain a single word of at least 1 and at most 5 lower-case letters.

Output the lexicographically first word of minimum length Brandon can come up with. If it is not possible to come up with a single word, output -1.

Sample Input 1

```
2
aba
bab
```

Output for Sample Input 1

```
abab
```

Sample Input 2

```
3
ab
bc
ca
```

Output for Sample Input 2

```
abca
```

Problem 6
Word Game (continued)

Sample Input 3

```
2
x
y
```

Output for Sample Input 3

```
-1
```

2024/2025 SOUTHERN CALIFORNIA REGIONAL
INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 7
N-Segment Display

Liquid Crystal Displays (LCDs) are popular on budget and single-purpose devices such as wristwatches, medical devices, and calculators. These devices display a limited number of characters, and compose their characters from a set of individual segments rather than pixels. Unfortunately, segments burn out over time, making the display ambiguous. Your team is to write a program that, given a set of characters and their segment definitions, checks readings for segments that might be burned out and determines the possible characters that are being displayed.

Input to your program has two sections: the characters and their segment definitions, followed by a number of readings. The first line of input contains N, the number of segments in a display, $1 \leq N \leq 26$. The second line contains C, the number of possible ASCII characters that can be displayed, $1 \leq C \leq 64$. The next C lines are the character definitions. Each character definition line starts with the character in column 1, then a sorted list of segment IDs that are turned on to display the character. The first display segment ID is labeled as *a*, the second as *b*, etc. For example, an 8-segment display uses segments *a* through *h*; a 25-segment display uses segments *a* through *y*. A character that has no segments turned on has no segment IDs, and will appear as the tilde '~' in column 1. Each character's segment definition is distinct from all other characters. A character will only be defined once.

The line following the character definitions contains K, the number of N-segment displays on the device, $1 \leq K \leq 7$. Displays are numbered from left to right. The next line contains R, the number of readings, $1 \leq R \leq 100$. The remaining R lines contain K comma-separated display readings, showing the segments that are on for that reading. It is possible that all segments in a given display are burned out. The device will never attempt to display undefined characters. Segments will either work (can be turned on or off), or be burned out (always off) across all readings for each independent display.

Figure 1 is a device with three 8-segment displays. The device is designed to show positive or negative whole numbers, the letters 'E' and 'R', and a blank character. The first sample input matches the device in Figure 1.

Given all the readings, determine all possible characters that each reading could represent. For each reading, print a line with all the possible characters for the first display, and as necessary print a space and all the possible characters for the second display, all the way up to the Kth display. For each display, sort the possible characters in ascending ASCII order.

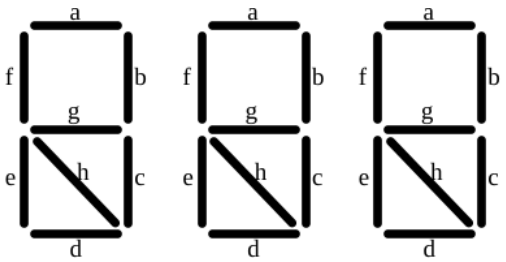

Figure 1. Three 8-segment displays, each with segments *a* through *h*.

Problem 7
N-Segment Display (continued)

Sample Input 1

```
8
14
0abcdef
1bc
2abdeg
3abcdg
4bcfg
5acdfg
6acdefg
7abc
8abcdefg
9abcdfg
-g
Eadefg
Rabefgh
~
3
4
adefg,abefg,abefgh
adefg,abcdef,bc
g,bcfg,abcdefg
,,bc
```

Output for Sample Input 1

```
68E R R
68E 0 1
- 4 8
1~ ~ 1
```

Sample Input 2

```
1
2
-a
~
1
2

a
```

Output for Sample Input 2

```
~
-
```

2024/2025 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 8
Free Solo

Alex is a climber who is planning his next free solo expedition. He wants to do a completely new climb this time, one that nobody else has done before. Before he starts the climb, he wants to get a rough idea of what the shortest path to the top is.

In climbing, Alex uses his hands and feet to hold on to "holds," which are points on the wall he is climbing where he can put either his hands or feet to stabilize himself.

In this simplified simulation of the route, Alex is modelled as a single point, from which 4 limbs of up to length 1 extend out from. Holds on the wall are also single points, and can be used by any of his limbs, but one hold can only be used by one limb at a time. In order to stay safe, Alex must ensure that at least 3 of his limbs are attached to holds at all times during the climb.

Alex starts at coordinate $(0, 0)$ on 3 holds located at $(-0.2, 0)$, $(0, 0)$, and $(0.2, 0)$ and has a target hold at (T_x, T_y) he is trying to reach. Find the length of the shortest path traced out by Alex's location until he is able to put one limb on the target hold. Alex's location is allowed to be anywhere on the wall as long as he stays safe, including locations with negative y-coordinates.

It is guaranteed that:

- No two holds will be within 10^{-3} units of each other.
- No two holds will be within 10^{-3} units of being distance 2 apart from each other.
- There will not exist a location on the wall where Alex is able to reach more than 6 holds.
- If Alex's reach increases or decreases by up to 10^{-6}, the length of the shortest path will not change by more than 10^{-5}.

The first line of input contains an integer n ($1 \leq n \leq 30$), the number of holds. The next n input lines contain two floating point numbers, the x and y coordinates of each hold ($-10 \leq x \leq 10, 0 \leq y \leq 10$). Floating point numbers will be expressed to exactly 5 decimal places.

The three starting holds are not included in the input. The target hold is the last hold given.

Output the length of the shortest safe path to the target hold, or -1 if no such path exists. Your answer will be accepted if the absolute or relative error is within 10^{-4} of the judge's reference answer.

Sample Input 1

```
1
0.00000 1.50000
```

Output for Sample Input 1

```
0.5000000000
```

Problem 8
Free Solo (continued)

Sample Input 2

```
1
0.00000 2.50000
```

Output for Sample Input 2

```
-1
```

Sample Input 3

```
4
0.00000 0.50000
-0.80000 1.50000
0.80000 1.50000
0.70000 2.20000
```

Output for Sample Input 3

```
1.3647093219
```

2024/2025 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 9
Letter Distance

You are given a string $a_1 a_2 \ldots a_n$ consisting of n upper-case letters of the English alphabet (A–Z). You want to calculate the following sum:

$$\sum_{i=1}^{n} \sum_{j=i}^{n} \left((j-i)^2 \cdot [a_i \neq a_j] \right)$$

where we define

$$[\mathit{condition}] = \begin{cases} 1, & \text{condition is true,} \\ 0, & \text{condition is false.} \end{cases}$$

Input to your program is a single line of at least 1 and not more than 10^6 upper-case English letters.

Print one line with the value of the above sum. Because the value of the sum can be very large, print the value modulo $(10^9 + 7)$.

Sample Input 1

```
ICPC
```

Output for Sample Input 1

```
16
```

Sample Input 2

```
SOCAL
```

Output for Sample Input 2

```
50
```

Sample Input 3

```
ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
```

Output for Sample Input 3

```
0
```

2024/2025 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 10
Birthday Cake

Given a birthday cake on the plane with four candles, decide whether two cuts with straight lines can cut the cake into four pieces, each with a single candle. The resulting pieces do not have to be of the same shape or size.

The input starts with a single integer T ($1 \leq T \leq 1000$), denoting the number of test cases. Each of the next T lines describes a single test case. Each test case contains eight integers, separated by spaces:

$$x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4$$

The integers denote the coordinates of the four candles. You are guaranteed that $-1000 \leq x_i \leq 1000$, $-1000 \leq y_i \leq 1000$, and that no two candles have the same coordinates. Candles are small enough to be treated as points. The cake may not be cut through a candle.

For each test case, output a single line containing "`Yes`" if the given cake can be cut into four pieces, each with a single candle on it. If it is impossible to do so, print "`No`" instead.

Sample Input

```
2
0 0 0 1 0 2 0 3
-5 0 5 3 2 10 -6 -3
```

Output for the Sample Input

```
No
Yes
```

2024/2025 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 11
Intuitive Elements

Brandon is learning the periodic table! However, he doesn't like some of the elements because the symbol of the element contains letters which are not present in the name of the element. He finds this to be unintuitive, especially because in other contexts, he expects abbreviations to not introduce random letters.

Given a string and a proposed abbreviation, determine if Brandon would find it intuitive. Brandon finds an abbreviation intuitive if and only if every letter that appears in the abbreviation appears in the original string. Brandon does not look at the abbreviation carefully, so it is acceptable for a letter to appear more times in the abbreviation than in the original string, or for the letters to appear in a different order between the string and the abbreviation.

The first line of input contains a single integer t ($1 \leq t \leq 10^3$). This is the number of test cases. Each test case consists of two lines. The first line contains a single string a of length at most 50. This string only contains lowercase letters. The second line contains a single string b that is strictly shorter than a and also only contains lowercase letters. Neither string will be empty.

For each test case, if all the letters in b appear in a, print a line containing the string "`YES`". Otherwise, print a line containing the string "`NO`".

Sample Input

```
4
magnesium
mg
silver
ag
aabb
bbb
aabb
ba
```

Output for the Sample Input

```
YES
NO
YES
YES
```

2024/2025 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 12
Lottery Confusion

The State of Confusion offers lottery games where players choose n distinct integers in a range from 1 to r inclusive, hoping to match the numbers drawn for the games each week. Games have different values for n and r in order to attract players—games with large values of n and r offer larger jackpots, while games with smaller values offer more frequent payouts.

The Confusion State Lottery is planning to offer a new way to play number combinations in quantity. Players can select numbers to be played by using question marks in place of digits when buying tickets. The question marks act as "wild cards"—by doing this, a player can purchase all the valid combinations that could be formed by substituting digits for the question marks. Valid combinations are those that result in distinct integers in increasing order from left-to-right, with all integers in the range 1 to r inclusive.

A lottery ticket has n selections. Each selection is either two digits, two question marks, or one digit and one question mark. Leading zeroes are used to select integers below ten.

Your team is to write a program that takes lottery ticket orders and determines the number of valid combinations on the ticket.

The input begins with a line containing a single integer T: the number of independent ticket orders, in the range 1 to 50 inclusive. Each ticket order starts with a line containing two space-separated integers: n, the number of integers a player is to select, in the range 1 to 9 inclusive; and r, the upper bound of the range of integers that the winning numbers are drawn from, in the range $n+1$ to 99 inclusive. The next line describes the n selections used for the ticket, separated by spaces.

For each ticket order, print a line with one value: the number of valid combinations represented on the ticket.

Sample Input

```
3
3 15
0? 10 1?
6 49
0? 15 19 1? 44 5?
6 60
0? 10 1? 2? 33 ?0
```

Output for the Sample Input

```
45
0
2430
```