Thirty-ninth Annual
University of Central Florida

High School Programming Tournament

Problems

Problem Name	Filename
Chaos and Disorder	chaos
Tyler Tyler	tyler
Apartment Subletting	sublet
Witches' Brew	witches
Meet in the Middle	meet
Avaricious Neologism	neologism
Can You See Me in the Dark?	dark
Land of Confusion	confusion
Halloween Doodle	doodle
Last Slime Standing	slime
Garden Flowers	flowers
Or Else	orelse

Call your program file: *filename*.c, *filename*.cpp, *filename*.java, or *filename*.py
Call your Java class: *filename*

For example, if you are solving Garden Flowers:
Call your program file: flowers.c, flowers.cpp, flowers.java, or flowers.py
Call your Java class: flowers

Notes

1. All solutions must read from standard input and write to standard output. The judges will ignore all output sent to standard error.

2. All input sets used by the judges will follow the input specification in each problem. You do not need to test for input that violates the input format specification.

3. Unless otherwise specified, specification for input and output that refers to "uppercase letters" means letters 'A' to 'Z' (inclusive) as they are used in English; input and output that refers to "lowercase letters" means 'a' to 'z' (inclusive) as they are used in English.

4. Submitted programs will be tested on additional judge cases and not just the sample cases given in the problem.

5. If you need to use the value for π, use the value 3.141592653589793.

6. If you want to know how to compute the absolute error, given the actual value of a quantity, x, and the measured value of a quantity, x_0, then the absolute error value, Δx, is calculated as:

$$\Delta x = |x_0 - x|$$

7. The relative error is defined as the ratio of the absolute error of the measurement to the actual measurement. If you want to know how to compute the relative error, given the actual value of a quantity, x, and the measured value of a quantity, x_0, then the relative error value, r, is calculated as:

$$r = \frac{|x_0 - x|}{x} = \frac{\Delta x}{x}$$

8. The modulo operation (often designated with the operator %) is defined as the remainder obtained after two operands are divided (e.g. 11 % 2 = 1, 29 % 3 = 2).

9. Images used are owned by either the original author, the University of Central Florida, or used under appropriate licenses (such as various Creative Commons licenses).

Chaos and Disorder

Filename: `chaos`

Samarth is an agent of chaos. He absolutely *despises* order. However, much to his dismay, some of his fellow peers on the programming team spend their free time sorting arrays. Samarth does his best to produce more disorder, of course, but it's getting hard for him to keep up, so he has enlisted you to help him!

The Problem:

Given a sorted array composed of n distinct integers ranging from 1 to n, reorder the elements in the array so that no element is in the place it was originally.

The Input:

The first line of input will consist of an integer, n ($2 \leq n \leq 100$), representing the length of the array. The second line of input will consist of n distinct space-separated integers, a_i ($1 \leq a_i \leq n$), each representing the i^{th} value in the array. The input array will be sorted in ascending order.

The Output:

Output a single line consisting of a single-space-separated list of integers representing a valid re-ordering of the input array such that no element is in the place it was originally. Any valid ordering will be accepted. The output for the sample cases is provided as examples; there may be other valid solutions.

Sample Input 1:

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

Sample Output 1:

```
5 3 4 1 2
```

Sample Input 2:

```
3
1 2 3
```

Sample Output 2:

```
2 3 1
```

Tyler Tyler

Filename: `tyler`

Andy's pattern recognition has gone out of control and he's started to see Tyler's name everywhere, from the words on his problemset to the variable names in his code, and even inside his brain! As many competitive programmers do, he wants to turn his current situation into a game: whenever he sees a string of letters, he will change it to a *Tyler* string. A *Tyler* string is a string that is composed of the word "tyler" repeated one or more times (such as "tyler", "tylertyler", "tylertylertyler", and so on). To do this, Andy can use three different operations as many times as he needs:

- Pick a letter in the string and change it to a different letter (ex: abcd → accd)
- Pick a letter in the string and remove it (ex: abcd → abd)
- Insert any letter at any position in the string, including at the beginning or end of the string (ex: abcd → axbcd or abcd → xabcd)

Andy also cares about efficiency, so he wants to determine the minimum number of operations to change a given string into a *Tyler* string. Help Andy figure out the most efficient way to do this.

The Problem:

Given a string, find the minimum number of operations to change it into a *Tyler* string.

The Input:

The first and only line of input is a single string, *s*, consisting of between 5 and 1,000 (inclusive) lowercase letters.

The Output:

Output a single integer: the minimum number of operations required to change the string into a *Tyler* string.

Sample Input 1:

```
abcdefghi
```

Sample Output 1:

```
8
```

Sample Input 2:

```
tyletler
```

Sample Output 2:

```
2
```

Sample Input 3:

```
tltrtm
```

Sample Output 3:

```
4
```

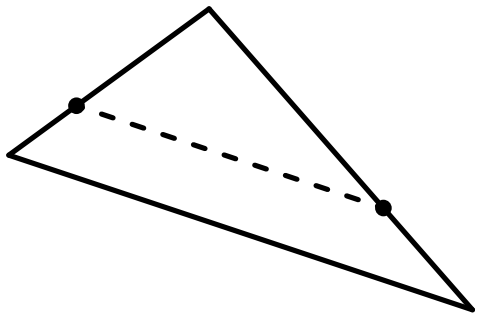
Apartment Subletting

Filename: `sublet`

Aria loves triangles. She loves them so much that she has found herself a completely triangular apartment! Unfortunately, the apartment is rather expensive (and quite a bit larger than she needs), so she is hoping to sublet part of the room for some extra cash. That is, she will split the room into two sections of equal area and rent out one of these sections.

Aria wants to divide her apartment into two rooms of equal area by building a single dividing wall. Of course, one of these rooms must be triangular to keep for herself. Aria has decided that this new wall must be parallel to one of the existing outer walls of the apartment, dividing the space into one triangular room and one trapezoidal room.

Aria has tasked you, the local Home-Space Partition Technician, with finding the minimum length of this dividing wall. Use your geometric prowess to determine this length, in order to minimize Aria's cost of building the wall.



The dashed line above represents a possible dividing wall for the first sample case. Note that this is not necessarily a solution with minimal length.

The Problem:

Given the three outer side lengths of Aria's apartment, find the minimal length of a dividing wall that is parallel to an existing side and divides the apartment into two rooms of equal area.

The Input:

The input will contain three integers, a, b and c ($1 \le a \le 10^6$; $1 \le b \le 10^6$; $1 \le c \le 10^6$), representing the lengths of the outer walls of Aria's apartment. respectively. It is guaranteed that the given side lengths form a triangle with positive area.

The Output:

Output a single real number: the minimum length of a dividing wall as described above. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Sample Input 1:

```
3 4 5
```

Sample Output 1:

```
2.121320
```

Sample Input 2:

```
4500 4500 4500
```

Sample Output 2:

```
3181.980515
```

Witches' Brew

Filename: `witches`

"Double, double toil and trouble;

Fire burn and cauldron bubble.

The witches' power may be great

But will it stand against Hecate?"

Sick of the witches dooming mortals with their prophetic visions, Hecate has placed the witches from Macbeth under a curse to never speak in rhymes again. This cannot stand, so the witches are making another brew in their cauldron: this time, a potion to break the curse.

However, the curse is quite dastardly and can only be broken if each of the three witches puts the exact same number of ingredients into the cauldron. Since the number of ingredients can be large, they have asked you to determine whether or not they will be able to break Hecate's curse.

The Problem:

Given the total number of ingredients the witches will collectively put in their cauldron, determine whether or not they can successfully make their potion.

The Input:

There is only one line of input, containing a single integer, x ($1 \leq x \leq 10^6$), representing the number of ingredients in the witches' brew.

The Output:

Output a single line. If the witches can successfully make their potion, output the text "`Something wicked!`" Otherwise, output text reading "`Toil and trouble.`"

Sample Input 1:	**Sample Output 1:**
`27`	`Something wicked!`

Sample Input 2:	**Sample Output 2:**
`101`	`Toil and trouble.`

Meet in the Middle

Filename: `meet`

Remus and Plinius are meeting up for a feast and they have decided to model a classic problem!

They don't particularly care where they eat, so they will first meet up and then find a suitable venue. Remus will set out from his home in Rome at the same time that Plinius sets out from Phoenicia; the two will move towards each other until they meet at the same location and then find a lunch spot nearby.

Unfortunately, there is now only one road that leads to Rome. Because of this, Remus' and Plinius' journeys can be modeled on the number line, with their starting positions represented by two points on the line.

Remus and Plinius each travel at their own constant speed. If the two set out at the same time, how long will it be before they meet at the same location?

The Problem:

Given the locations of Rome and Phoenicia on the number line, as well as Remus' and Plinius' travel speeds, find the amount of time it will take for Remus and Plinius to meet up at the same location.

The Input:

The first line of input contains two integers, x_r and x_p ($0 \le x_r < x_p \le 10^8$), representing the positions on the number line of Remus' and Plenius' starting points, respectively.

The second line contains two integers, s_r and s_p ($1 \le s_r \le 10^8$; $1 \le s_p \le 10^8$), representing the travel speed (in units per second) of Remus and Plinius, respectively.

The Output:

Output a single real number: the number of seconds it will take for Remus and Plinius to meet up. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-4}.

Sample Input 1: **Sample Output 1:**

`0 12` `1 1`	`6.000000`

Sample Input 2: **Sample Output 2:**

`267 100000` `13403 18343`	`3.141592`

Avaricious Neologism

Filename: `neologism`

The University of Central Florida, publisher of the Central Florida English Dictionary, has hired you to invent some new words for their next edition. You get paid by the letter, so ideally you would like to create the longest word you can. However, you can't just use any random arrangement of letters, since in English, some pairs of letters (also known as bigrams) aren't allowed to be adjacent in a word (such as "cf" or "hk").

UCF has given you a list of pairs of letters that are allowed to appear next to each other, in that order, in a valid word. Your goal is to construct the longest word possible.

The Problem:

Given a list of allowed bigrams, construct a longest word possible (if one exists) or indicate that an infinitely long word can be constructed. For a word to be allowed, every pair of adjacent characters in the word must appear in the list of allowed bigrams.

The Input:

The first line of input contains a single integer, n ($1 \leq n \leq 26^2$), representing the number of allowed bigrams. The next n lines each contain a string s consisting only of 2 lowercase letters, representing a bigram that is allowed to appear in the word. It is guaranteed that all given bigrams are unique.

The Output:

The first line of output should be a single integer, l, representing the longest possible length of a valid word, or -1 if that length would be infinite. If $l \neq -1$, the second line of output should be a single string of length l, representing a valid word of maximal length. If there are multiple answers, any valid answer will be accepted.

Sample Input 1:

```
4
sp
ht
pt
hs
```

Sample Output 1:

```
4
hspt
```

Sample Input 2:

```
1
aa
```

Sample Output 2:

```
-1
```

Can You See Me in the Dark?

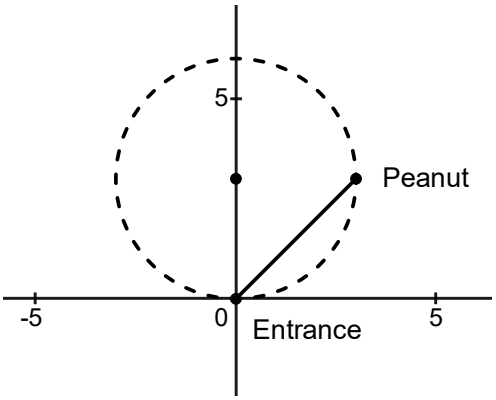
Filename: `dark`

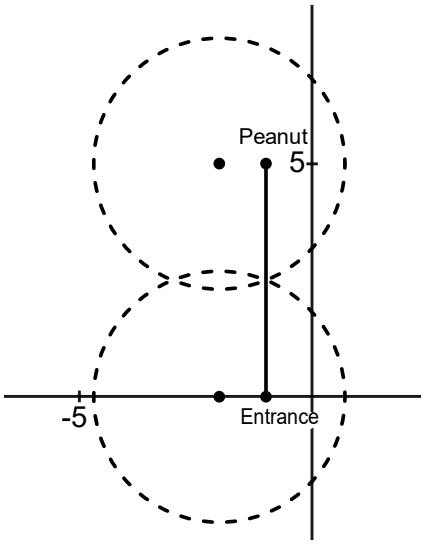
After the Elephant Espionage Brigade's successful deployment of the Hyper Soft-Pitch Transmitters, they have finally gathered some highly confidential intel: the location of the Golden Peanut!

To capture the Golden Peanut, the elephants have once again deployed their most elite spies: Ellie, Elliot, and Ellanah. While traversing the many obstacles that blocked their path, the three spies prepared to enter the Hidden Stronghold of Peanut Treasures. Despite the lack of traps in this room, there was still one major barrier to their success: security cameras.

To circumvent the security cameras, Ellanah, the brigade's technology guru, has added dark zones into the room. Each dark zone prevents all points within a certain distance from its center from being seen by any cameras. Since these dark zones can be quite unstable, the elephants want to minimize the amount of time they spend in the room. Therefore, they will take a straight line path to the Golden Peanut and back to the door. Luckily, the elephants can hide within or on the border of the zones!

Ellanah can control the size of the dark zones as long as they all maintain the same radius, but she notices that the larger she makes them, the more unstable they become. She has asked you to help her determine the smallest radius the zones can have such that Ellie's and Elliot's path is fully covered by the dark zones.





The images above show the solutions to the two sample cases provided.

The Problem:

Given the center location of each dark zone, along with the locations of the entrance and the Golden Peanut, determine the minimum radius needed such that Ellie's and Elliot's straight line path between the entrance and the Golden Peanut is fully covered by dark zones.

The Input:

The first line of input will contain a single integer, n ($1 \le n \le 10^5$), representing the number of dark zones Ellanah has created. The following n lines will each contain two integers, x_i and y_i ($-10^5 \le x_i \le 10^5$; $-10^5 \le y_i \le 10^5$), representing the location of the center of the i^{th} dark zone.

The final line of input will contain four integers, x_e, y_e, x_p and y_p ($-10^5 \le x_e \le 10^5$; $-10^5 \le y_e \le -10^5$; $-10^5 \le x_p \le 10^5$; $-10^5 \le y_p \le 10^5$), representing the locations of the entrance and the Golden Peanut, respectively. The entrance and the Golden Peanut are guaranteed to not be at the same location.

The Output:

Output a single real number: the minimum radius needed to fully cover the straight line path between the entrance and the Golden Peanut. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-4}.

Sample Input 1:

```
1
0 3
0 0 3 3
```

Sample Output 1:

```
3.00000000
```

Sample Input 2:

```
2
-2 0
-2 5
-1 0 -1 5
```

Sample Output 2:

```
2.69258240
```

Land of Confusion

Filename: `confusion`

In a remote region of the magical realm of Lexia, there exists the mystical *Land of Confusion*. Covered in the Hazy Smoke of Puzzling Turmoil, not much is known about the people of this land beyond fables of their puzzling riddles, perplexing mazes, and love of tongue twisters.

Legend has it that the only way to gain entrance to this mysterious land is to impress the Titan of Tongue Twisters, Thestra. Thestra loves crafting extravagant tongue twisters to test the minds of those attempting to enter the Land of Confusion, and any traveler attempting to enter must successfully complete Thestra's tongue twister test.

You and a group of travelers plan to venture to the Land of Confusion, so you want to train in the trade of tongue twisters. The travelers have crafted several sentences and want you to verify that they are truly tongue twisters. In order to do this, you want to check if every word in the sentence starts with the same letter.

The Problem:

Given the sentence the travelers created, determine whether it is a valid tongue twister by checking if every word begins with the same letter.

The Input:

The first line of input will consist of a single integer, n ($2 \leq n \leq 10$), denoting the number of words in the sentence. The next line will consist of n single-space-separated words, representing the potential tongue twister. Each word will consist of at most 10 lowercase letters.

The Output:

If the sentence is a valid tongue twister, output "`Not even Thestra can compete!`" Otherwise, output "`Thestra is disappointed!`"

Sample Input 1:

```
3
tricky tongue twisters
```

Sample Output 1:

```
Not even Thestra can compete!
```

Sample Input 2:

```
2
invalid test
```

Sample Output 2:

```
Thestra is disappointed!
```

Halloween Doodle

Filename: `doodle`

Sathvik is a big fan of the Magic Cat Academy series, a selection of browser games released every four years as part of the Halloween Google Doodle. In the game, you play as a cat with a magic wand. Enemies are coming from all directions to attack you and each enemy has a string of shapes above their head; to eliminate an enemy, you must empty their string. Every time you draw a shape with your magic wand, one occurrence of that shape will be removed from the beginning of every enemy's string if it is the first element of that string.

Being analytically inclined, Sathvik wants to analyze some possible games. Given the moves he's planning to make, he wants to know how many enemies he will successfully eliminate. Can you help him figure it out?

The Problem:

Given a list of the enemies' strings and a string representing the moves Sathvik is planning to make, determine how many enemies will be eliminated.

The Input:

The first line of input will be a string of length between 1 and 10^6 lowercase letters, representing the moves Sathvik is planning to make. The second line of input will consist of an integer, n ($1 \le n \le 10^5$), representing the number of enemies. The next n lines of input will each consist of a string of lowercase letters representing an enemy. The total length of all enemy strings is guaranteed not to exceed 10^6.

The Output:

Output a single integer representing the number of enemies Sathvik will eliminate after making all his planned moves.

Sample Input 1:

```
hweolrld
2
hello
world
```

Sample Output 1:

```
1
```

Sample Input 2:

```
aabcde
3
aaec
abdc
da
```

Sample Output 2:

```
0
```

Last Slime Standing

Filename: `slime`

Welcome to the Heavy Slime Power Tournament, where many slimes enter, but only one may leave! We have gathered n brave slimes who will battle it out to see who will be the last slime standing! The tournament will consist of n - 1 rounds. In each round, two slimes will be *arbitrarily* selected to do battle. Slime battles are very predictable; when two slimes battle, the heavier one will always win. If both slimes have the same mass, then they are both equally likely to be declared the winner. After two slimes battle, the loser will be merged (humanely) with the winner, adding the loser's mass to the winning slime. The slime remaining after all n - 1 rounds have completed is declared the winner.

You have noticed that, due to the nature of the tournament, many different slimes have a chance of winning the tournament. Before you decide which slime to root for, you want to determine which slimes have a chance of winning in at least one possible tournament sequence.

The Problem:

Given the mass of each slime entering the tournament, determine for each slime if it can win the tournament in at least one sequence of battles that could comprise a tournament.

The Input:

The first line of input contains a single integer, n ($1 \leq n \leq 10^5$), representing the number of slimes entering the tournament. The following line contains n single-space-separated integers, each representing the mass of a slime (in standard goo units). Each slime weighs between 1 and 10^9 (inclusive) standard goo units.

The Output:

Output a line consisting of a single string of n characters. The i^{th} character of the string should be '`W`' if it is possible for the i^{th} slime to win the tournament and '`L`' otherwise.

Sample Input 1:

```
5
7 1 3 10 2
```

Sample Output 1:

```
WLLWL
```

Sample Input 2:

```
3
7 7 7
```

Sample Output 2:

```
WWW
```

Garden Flowers

Filename: `flowers`

Thomas has decided to abandon his career as a computer scientist and has taken up gardening. With great care, he has cultivated a garden in the shape of an n by n grid. Each plot in the garden contains several flowers of a single type, where the type is represented by an uppercase letter from 'A' to 'Z'. Thomas wants to take a daily stroll through the garden, starting from the top-left plot and ending at the bottom-right plot.

During each stroll, Thomas can only move to directly adjacent plots, either by moving down or to the right. At each plot he visits, he collects exactly one flower from the plot and adds it to a necklace. The flowers are collected in the order that Thomas visits the plots along his path.

Thomas wonders if two different strolls (i.e. paths) can result in the same necklace, where the necklace is the sequence of flowers collected along the path.

Your task is to determine whether there exist at least two different paths through the garden that produce the same necklace. Two paths are different if there is a plot that is visited in one of the paths but not the other.

The Problem:

Given a grid of uppercase letters representing Thomas' garden, determine whether there exists two different strolls through the garden that produce the same necklace.

The Input:

The first line will contain a single integer, n ($2 \leq n \leq 100$), representing the width and height of the garden. The following n lines will each contain n uppercase letters ('`A`' to '`Z`') representing the rows of the flower garden. The rows are given in order from top to the bottom; each row is given from left to right.

The Output:

Output a single line consisting of the string "`unique`" if no two paths through the flower garden produce the same necklace. Otherwise, output "`duplicates`" instead.

Sample Input 1:

```
3
ROS
OSE
CKS
```

Sample Output 1:

```
duplicates
```

Sample Input 2:

```
2
HS
PT
```

Sample Output 2:

```
unique
```

Or Else

Filename: `orelse`

The HSPT judges are celebrating yet another successful contest! Once again, Glenn has decided to reward the judges with a party. This year, Glenn is going all out and planning a party with a wide variety of foods beyond the traditional pizza party of previous years. However, the judges are picky eaters and have provided specific requests for the menu.

Each judge has given Glenn a preference of the form "*A* or *B* else *C*". They may also negate any of these dishes, meaning they might prefer for a dish *not* to be served. In that scenario, the name of the dish will be preceded by '!'. Each judge will be satisfied if their preferences on *A* or *B* are satisfied, but if neither *A* nor *B* are satisfied, the judge will only be satisfied if *C* is. Equivalently, the judge will be satisfied if any of the preferences are satisfied.

Glenn needs your help! He must determine if it's possible to satisfy *all* of the judges' preferences with a single menu. Can you find a way to create a menu that ensures that every judge is happy, or is it impossible to meet all their expectations at once?

The Problem:

Given a list of the judges' preferences for food on the menu, determine if there is a menu that satisfies all of the judges by meeting at least one of each judge's preferences.

The Input:

The first line of input contains a single integer, m ($1 \le m \le 50$), representing the number of judges with requests for the menu. The next m lines are each of the form "`A or B else C`" where *A*, *B*, and *C* are each a string outlining whether a judge has a positive or negative preference for a dish. If the judge has a positive preference, the string will be composed of only the dish name. If the judge has a negative preference, the string will be composed of a '`!`' followed by the dish name. The dish name will be a string of between 1 and 10 (inclusive) uppercase letters. Note that *A*, *B*, and *C* can refer to the same dish. It is guaranteed that at most 15 unique dishes are mentioned in the judges' preference.

The Output:

First, output a line containing either "`Dinner is served!`" if there is a menu that satisfies the judges or "`Better luck next year!`" if there is not a menu satisfying the judges. If there is a valid menu, then output a second line with a single integer, d ($0 \le d \le 15$), representing the number of dishes on the valid menu. Then, output d lines, each containing the name of a dish that is on the menu and as given in the input. If there are multiple possible menus, output any valid menu.

Sample Input 1:

```
2
PIZZA or SUSHI else !SALAD
!SUSHI or !PIZZA else SALAD
```

Sample Output 1:

```
Dinner is served!
1
PIZZA
```

Sample Input 2:

```
1
CAKE or !CAKE else COOKIE
```

Sample Output 2:

```
Dinner is served!
0
```

Sample Input 3:

```
2
PIZZA or PIZZA else PIZZA
!PIZZA or !PIZZA else !PIZZA
```

Sample Output 3:

```
Better luck next year!
```