BRAINIACS OLYMPIAD

CODING SAMPLE CHALLENGES

info@brainiacsolympiad.com
www.brainiacsolympiad.com

©BRAINIACS OLYMPIAD
ORGANIZED BY MINDCSAPE INTERNATIONAL LLC

BRAINIACS OLYMPIAD 2026
CODING SAMPLE CHALLENGES (GRADES 10–12)

Easy challenges – 20
Medium challenges – 20
Hard challenges – 10

Easy challenges

1. Frozen railcars

A cargo train hauling tons of ice cream derailed in the northern tundra, covering the area in a giant frozen dessert landscape. While crews work to clear the mess, all other trains are diverted to an old icy branch line where they must travel slowly and make two safety stops. Railway control needs your help to calculate travel times under these conditions.

Task

Write a program that calculates the total travel time, in minutes, for a train given: the number of miles it needs to travel on the icy backup route, and its constant speed (in miles per hour). Every train also makes two safety stops, each lasting 5 minutes.

Input

A single line with two floating-point numbers - the number of miles traveled and the number of minutes elapsed.

```
40 20
```

Output

Print the total travel time in minutes, including the two mandatory 5-minute stops.

```
130
```

Explanation

Distance: 40 miles
Speed: 20 mph → 40 miles / 20 mph = 2 hours = 120 minutes
Two safety stops: 5 + 5 = 10 minutes
Total time: 120 + 10 = 130 minutes

Additional examples:

Input	Output
10 40	25
75 25	190
5 10	40

2. Halfway floor

David always takes the stairs and notices which floor is exactly halfway to his destination. Given a floor number, he wants to know the floor he is actually climbing to.

Task

Write a program that calculates Davids's destination floor given k, the floor that is exactly halfway through his ascent. The input is a single natural number k (k < 100). Output the floor David is climbing to.

Input: 2

Output: 3

Additional examples:

Input	Output
1	1
10	19
11	21
98	195

3. Cosmic Delivery

In the Interstellar Delivery Challenge, each team of pilots must launch a cargo pod through a hyper-dimensional portal. The pod cannot have any propulsion system, so it travels solely based on the initial speed and trajectory. The winner is the team whose pod reaches the portal first.

Task

Write a program that determines which team delivers its pod first, given the distance to the portal and the pod's speed for each team.

Input

The first line contains an integer n ($1 \le n \le 7$) — the number of teams whose pods are aimed correctly at the portal. Each of the next n lines contains **Distance** in kilometers and **Speed** in kilometers per second.

```
3
Dragons 914.8 168.2
ChartMonkeys 1273.1 193.5
SolarWind 1144.6 181.3
```

Output

Print the name of the winning team followed by the time in seconds (as a floating-point number) it takes for the pod to reach the portal.

```
Dragons 5.438763377
```

Additional examples:

Input	Output
2 StarRiders 512.5 48.2 MoonHawks 601.3 59.7	MoonHawks 10.0720268

4. Electric charges

You need to calculate the total amount of an electric bill. The first 1000 kilowatt-hours (kWh) are billed at 7.63 cents per kWh, and any usage above 1000 kWh is billed at 9.25 cents per kWh.

Task

Write a program that calculates the total amount owed in dollars for a given number of kilowatt-hours used.

Input

The input will be a single integer greater than zero, representing the kilowatt-hours used.

```
1500
```

Output

Print the total amount owed in dollars with up to two decimal places:

```
122.55
```

Additional examples:

Input	Output
900	68,67
1700	141,05

5. Fun with commas

In this problem, you must separate each word in a list with commas, except for the last word, which should be preceded by the word “and”. Each word will be entered on the console on its own line. The list will have at least one word and a maximum of fifty words, and it will end with the word “quit”.

Task

Write a program that reads a list of words and prints them in a single line, separated by commas, with “and” before the last word.

Input

Prompt for the input as shown below. Enter a list of words of various lengths ending with the word quit. The list will have a minimum of one word and a maximum of fifty words. Data will be entered from console. You do not need to edit the input data. Rerun your application to test each test case.

```
Enter next word or quit: one
Enter next word or quit: two
Enter next word or quit: three
Enter next word or quit: quit
```

Output

The list of words with each pair of words separated by a comma except for the last. The last pair of words are separated by the word "and".

```
one, two and three
```

Additional examples:

Input	Output	Input	Output
1	1 and 2	one	one
2		quit	
quit			

6. Treasure Division

Pirate Captain Ruby has collected a series of treasure chests, each containing a certain number of gold coins. She wants to know the largest number of times she could split the coins in one chest by the coins in another chest without considering fractions. That is, she wants the largest whole number result of dividing any chest’s coins by another chest’s coins.

Task

Write a program that reads a sequence of integers representing the number of coins in each chest (ending with 0) and calculates the largest integer quotient possible by dividing one chest’s coins by another. Ignore the 0 at the end.

Input

```
4 5 33 20 19 0
```

Output

Print a single integer — the largest whole-number quotient possible.

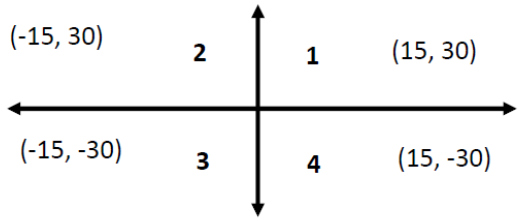
```
8
```

Additional examples

```
Input              Output
6 2 15 9 0         7
10 3 5 8 0         3
```

7. Rocket Navigation

A space probe is flying across a 2D map of a planet. The map is divided into four zones:
Zone 1: X > 0, Y > 0
Zone 2: X < 0, Y > 0
Zone 3: X < 0, Y < 0
Zone 4: X > 0, Y < 0
The probe's instruments provide the coordinates (X, Y) of its current location. Your task is to tell which zone the probe is in.



Task
Write a program that reads the X and Y coordinates of the probe and outputs the zone number.

Input
Prompt for the input as shown below. Enter two integers separated by a space. Neither number can be zero.

```
Enter X and Y separated by a space: 45 90
```

Output
Output the quadrant number (1, 2, 3 or 4) for the point. Use the format shown below.

```
Zone 1
```

Additional examples

Input	Output
`Enter X and Y separated by a space: -15 20`	`Zone 2`
`Enter X and Y separated by a space: -12 -8`	`Zone 3`

8. Less or More

In this problem you must examine a list of numbers and determine how many times a number in the list has a lower number to its "right". For example, in the list 3, 9, 4, 6, 7, 5, there are 3 cases where a number in the list has at least one lower number to the right. The number 9 has the number 4 to its right. The number 6 has the number 5 to its right. The number 7 has the number 5 to its right.

Input
Prompt for the input as shown below. Enter an integer, n, that represents the count of numbers to follow. Enter n integers with each separated by a space. Each integer will be greater than 0.

```
Enter count, integers separated by a space:
6 28 7 6 21 88 89
```

Output
A count of how many times a number in the list has a lower number to its right.

```
2
```

Additional examples

Input	Output
5 10 5 3 11 8	3
4 1 2 3 4	0

9. Letter Frequency

In this problem, you are given a sequence of letters (both uppercase and lowercase). Your task is to determine how many times each letter appears, counting uppercase and lowercase versions as the same letter. For example, A and a are treated as the same letter.

Task
Write a program that reads a sequence of letters ending with the special character # and outputs each letter and how many times it appears. The output should be in alphabetical order. Use the words “time” for one occurrence and “times” for multiple occurrences.

Input

```
Enter your letters separated by a space
a B b A C c a #
```

Output

```
a occurs 3 times
b occurs 2 times
c occurs 2 times
```

10. Tree Growth

In a virtual forest, a tree’s name determines its growth speed. Each letter in the tree’s name is assigned a value as follows:

- Vowels (a, e, i, o, u) are 5 points each.
- Special consonants (b, c, d, f, g) are 3 points each.
- All other consonants are 2 points each.

The sum of these points determines the tree’s growth speed as follows:

Sum Points	**Growth speed**
Less than 25	Slow
25 to 35	Medium
Greater than 35	Fast

Task
Write a program which reads in the name of the tree and outputs the tree’s growth speed.

Examples:
Input: baobab
Output: Slow
Explanation: The sum points of baobab is 3 + 5 + 5 + 3 + 5 + 3 = 24. Since 24 is less than 25 the growth speed is Slow.

Input: pineapple
Output: Medium

Explanation: The sum points of pineapple is 2 + 5 + 2 + 5 + 5 + 2 + 2 + 2 + 5 = 30. Since 30 is in the range 25 to 35 the growth speed is Medium.

Test your program with the following cases:
2.a. oak
2.b. magnificent
2.c. sequoia
2.d. yellowwood

11. Alien Radar: Circle Defense

Your spaceship is patrolling a sector of space, and enemy drones are appearing on your radar. To defend your base, you need to label each drone:

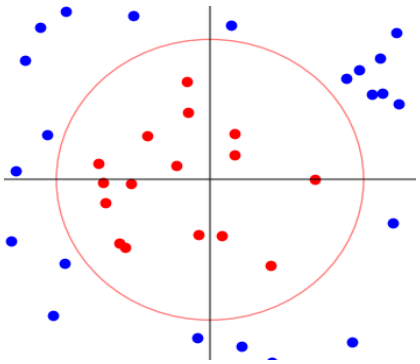
- **RED** → Drone is safely within the protective shield (inside or on the circle).
- **BLUE** → Drone is outside the shield and must be intercepted.

The shield is **circular**, centered at (0,0) with a **radius of 1**. The distance of a drone (x, y) from the center is:

$$\text{distance} = \sqrt{x^2 + y^2}$$

If the distance is **greater than 1**, the drone is outside the shield (BLUE). Otherwise, it is protected (RED).

Task
Given the point (x, y), write a program that outputs BLUE if the point lies outside the circle and outputs RED if the point lies inside or on the circle.
The first line of input contains the coordinates x and y.



Examples
Input: 0.1 0.3 Output: RED
Input: 0.9 0.53 Output: BLUE

Test your program with the following cases:
a) 0.9 0.8
b) 0.2 0.96
c) -0.5 0.87
d) 0.83 -0.55

12. Triangle Types

Sipho needs your help distinguishing between different types of triangles. Given three angles, given as three positive integers in degrees, you must first determine if they constitute the angles of a triangle. Note that the angles in a triangle must add up to 180°. If the angles form a triangle, you must then determine if it is an equilateral, isosceles, or scalene triangle.
Write a program that, when given three angles, determines what type of triangle they make. If no triangle can be made with the given angles, output "IMPOSSIBLE". Otherwise, if it makes an equilateral, isosceles, or scalene triangle, output "EQUILATERAL", "ISOSCELES", or "SCALENE" respectively (without quotation marks).

Note: An equilateral triangle has all three angles equal, an isosceles triangle has only two angles equal and a scalene triangle has three different angles.

Examples:
Input: 30 60 70 Output:IMPOSSIBLE

Input: 40 40 100 Output:ISOSCELES
Input: 50 60 70 Output:SCALENE

Test your program with the following cases:
a) 60 60 60

b) 80 20 80

c) 75 34 12

d) 79 23 78

13. Fibonacci
In the Fibonacci sequence every number, from the third number onwards is the sum of the previous two numbers. For example: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89 etc.
Write a program that will print out a Fibonacci sequence of length L that starts with the number F and S (First and Second). The numbers must be separated by commas and spaces.
F and S will be whole numbers between -100 and 100. L will be a positive whole number less than 100.
Input:
F = 2
S = 3
L = 8
Output: 2, 3, 5, 8, 13, 21, 34, 55

Additional examples:
F=2, S=2, L=10
F=4, S=6, L=8
F=-2, S=3, L=12

14. Hidden Signal Locator
In a data sequence, a single character signal is hidden among other characters.
The goal is to determine the **position** of this special character within the sequence.

Input

Line 1: A single character C — the signal to locate.

Line 2: A string S of length 10 to 1000, containing exactly one occurrence of C.
AAAAAAAAZAAAAAAAA

Output
Print the zero-based index of the signal in the format:

Z found at index 8

15. Sum of Series
Write a program to determine the sum of the first n terms of the following series:
1/2 + 2/3 + 3/4 + 4/5 + 5/6 + …, where n is between 1 and 10000. Output the answer rounded off to 2 decimal places.
Note: A value between 0.005 and 0.009 rounds off to 0.01, while a value between 0.001 and 0.004 rounds off to 0.00. For this question only, your program does not have to do the rounding; you may round off by hand.

Example: n = 4
Answer: 2.72 (rounded off from 2.716666)
Test your program with the following cases:
Give your answer as a number only with no spaces, e.g. 1234
a) n = 2
b) n = 10
c) n = 9999
d) n = 10000

16. Prime Numbers
Prime numbers form a fundamental part of mathematics. Recall that a prime number is a positive integer greater than 1 and which is divisible only by itself and 1. The first five prime numbers are 2, 3, 5, 7 and 11.

Given a positive integer ��, write a program which outputs the ��-th prime number. n
Examples:
Input: 3 Output: 5
Input: 10 Output: 29
Input: 15 Output: 47

Test your program with the following cases: Give your answer as a number only with no spaces, e.g. 1234
a) 30
b) 101
c) 12321

17. Robot Step Counter
A delivery robot moves inside a warehouse on a straight path.
Each movement command is either:
F – move forward 1 meter
B – move backward 1 meter
Your job is to determine the robot's final position relative to its starting point.

Task
Write a program that reads a string consisting of the letters F and B only.
Calculate the final displacement:
Forward steps count as +1
Backward steps count as –1
Then print the final position.

Input
FBFFBFBF

Output
A single integer representing the net position.
2

Additional exmaples

Input	Output
FFFFBFB	3
BBBBFF	-2
FBFBFBFB	0

18. Tax Detective

Johnny loves to travel and explore the world's markets. He notices that in most countries, the prices of items already include taxes, unlike in the U.S. This makes him curious: how much of what he pays is actually tax?

Your task is to help Johnny calculate the tax portion for each of his purchases so he can keep track of his spending and taxes separately.

Task

For each purchase, you will receive the total amount (which includes tax) and the local tax rate as a percentage. Your program must calculate the tax amount included in the purchase using the following formulas:

Calculate the pretax price (the price before tax):

pretax_price = total / (1 + tax_rate / 100)

Calculate the tax amount:

tax_amount = total - pretax_price

The tax amount should not be rounded up, and all monetary values should be displayed with two decimal places.

Input

The first line contains a single integer N, representing the number of purchases to process.

The next N lines each contain two numbers separated by a space:

tax_rate – the tax rate as a percentage (0 < tax_rate < 100).

total – the total purchase amount including tax (0 < total < 10,000).

```
5
8.25 500.00
5.00 17.35
2.37 412.13
6.00 99.99
7.50 77.77
```

Output

```
On your $500.00 purchase, the tax amount was $38.11
On your $17.35 purchase, the tax amount was $0.83
On your $412.13 purchase, the tax amount was $9.54
On your $99.99 purchase, the tax amount was $5.66
On your $77.77 purchase, the tax amount was $5.43
```

19. Party Handshakes

You are at a lively party where several couples have arrived. As part of the social protocol, everyone introduces themselves and shakes hands with others. However, no one shakes hands with themselves or with their own partner.

You are curious: how many handshakes happen in total?

Task

Write a program that calculates the total number of handshakes at the party.

Each person shakes hands once with every other person who is not themselves or their partner.

The input will be the number of couples attending the party.

Input

A single integer N (1 ≤ N ≤ 100), the number of couples at the party.

4

Output
A single integer: the total number of handshakes that occur.
24

Explanation
Couple AA: shakes hands with BB, CC, DD → 6 handshakes
Couple BB: shakes hands with CC, DD → 4 handshakes
Couple CC: shakes hands with DD → 2 handshakes
Couple DD: no new handshakes left → 0
Total: 6 + 4 + 2 + 0 = 12 handshakes per person counted twice → 24 total

20. Solar Panel Efficiency
A research team is testing new solar panels in different weather conditions.
Each panel produces a certain amount of energy (in watts). But due to dust, clouds, and temperature changes, some readings are negative — these represent faulty measurements and must be ignored.

Task
Write a program that reads a sequence of integer watt values.
Stop reading when the value -999 is encountered (this end value should not be counted).
Only positive numbers represent valid energy readings.
Your program must calculate:
- the total energy (sum of all positive readings)
- the number of valid readings
- the average output, rounded down to the nearest whole number (If no valid readings, print 0.)

Input
50 -3 20 -10 40 -999

Output
Total energy: 110
Valid readings: 3
Average output: 36

Medium challenges

1. It Adds Up
In this problem you will be given a target number and a list of numbers to search. You will need to identify all pairs of numbers that sum to the target number. For example, given the target number of 6 and the list of 16, 11, 2, 8, 6, 7, 4, 33, -2, the pairs (2, 4) and (8, -2) sum to 6. In addition to listing the pairs that sum to the target number, you will need to list their position in the original list. The pair (2, 4) was located at 3 and 7. The pair (8, -2) was located at 4 and 9.

Input
Prompt for the following integer values. The first integer value will be a positive or negative integer or zero and will represent the target sum. The second integer is the number of numbers (N) in the list. This is followed by N positive or negative integers, or zero. Each integer value will be separated by a space. Data will be entered from the console. You do not need to edit the input data. Rerun your application to test each test case.
Enter the desired sum, number of elements, each element:
6 9 16 11 2 8 6 7 4 33 -2

Output
Display each pair of numbers that sum to the target number formatted as shown below. This is followed by the location of each number formatted as shown below. Display “No pairs found” if there are no pairs that sum to the target number.
(2, 4) found at [3] [7]
(8, -2) found at [4] [9]

2. Letter Builder
You are given a collection of letters and a target sentence. Your job is to check if the target sentence can be created using only the letters from the collection.
Rules:
Each letter in the collection can be used only once.
Spaces are ignored.
All letters are lowercase.
No punctuation is used in either the collection or the sentence.

Task
Write a program that reads a collection of letters and a target sentence, and then determines whether the sentence can be formed from the letters.

Input
First line: A string containing the collection of letters.
Second line: A string containing the target sentence.

```
friend icing on the sung
coding is fun
```

Output
Print possible if the sentence can be formed.
Print not possible if it cannot be formed.

```
possible
```

Additional examples
Input

```
please bake umpteen cookies
bugs are problems
```

Output

```
not possible
```

3. Potion Hunt in Excel Land
Wizard Calc is on a quest to collect magic potions hidden in Excel Land, where every cell contains a special potion. Each cell is labeled in Excel style: letters for columns and numbers for rows.

Columns 1–26 → A to Z
Columns 27–52 → AA to AZ
Columns 53–78 → BA to BZ
Columns 703–728 → AAA to AAZ
…and so on.

Wizard Calc receives a list of coordinates in row-column format and needs your help to find the exact cells that contain potions.

Task
Write a program that converts a row and column number into an Excel-style cell name, so Wizard Calc can collect the magic potions.

Input
Each line contains a row number r and a column number c in the format:
rXcY
where X is the row number (1–1,048,576) and Y is the column number (1–16,384).
The hunt ends when r0c0 is entered (do not process this line).

```
r1c1
r3c1
r5c26
r2999999c27
r52c52
r53c17578
r53c17602
r0c0
```

Output
Print the cell name for each coordinate so Wizard Calc can collect the magic potion.

```
A1
A3
Z5
AA2999999
AZ52
YZB53
YZZ53
```

4. TRANSFORMATION

Some doors can be unlocked using a key card – a plastic card with holes in it. A key card may be square like this: (O means a hole, X means no hole); but it may also be a rectangle or a triangle.

XXXO
XXOX
XOXX
XXOX

However in order to work, these specific key cards have to be rotated or flipped according to one or more of these transformation instructions:
R = Rotate 90° to the right
T = Top to bottom flip: you see the back – upside down
L = Left to right flip: you see the back – mirrored

Task:
Write a program that will take the given key card, rotate, and/or flip it as instructed, and print out the resulting key card. No card will be more than 40 by 40 spaces.

Example:
Input:
XXXO
XXOX
XOXX
XXOX
Transformation: R, L

Output:
XXXX
XXOX
XOXO
OXXX

Explanation:
After rotating by 90°
XXXX
XOXX
OXOX
XXXO

After a left to right flip:
XXXX
XXOX
XOXO
OXXX

Test your program with

(a) XXXO
XXOX
XOXX
XXOX
Transformation: T

(b) XXXO
XXOX
XOXX
XXOX
XXXX
Transformation: T, L, R

(c) O
XXO
OXXXO
XOOOXXX
Transformation: R, T, L

5. ANAGRAM
An anagram of a word contains the same letters as the original word, but in a different order.
Write a program that will calculate how many anagrams can be made with a given word – not counting the word itself. Anagrams that have the same letter-order may only be counted once. The words given will always be lower case.

Example 1:
Input: cat
Output: 5
[Explanation: the accepted anagrams for “cat” are: cta, act, atc, tac, tca. A total of 5]

Example 2:
Input: see
Output: 2
[Explanation: the accepted anagrams for see are: ese, ees. Although “ees” can be made up in two ways

by rearranging the letters of “see” , it only counts as a single anagram, because the final word is the same. A total of 2]

Use the following words to test your program
a: hound
b: foxhound
c: bookkeeper

6. Wizard Spell Combos

Introduction:
In Wizard Duel, wizards cast spells using four elemental runes: F, W, E, A.
Pressing a rune once casts a basic spell.
Pressing the same rune twice in a row casts a powerful spell.

For example:
F → Firebolt
FF → Fireblast (powerful spell)
Wizards can cast long sequences of spells, like FWFFEA. The duel system cannot read the raw sequence; it uses an encoded version:
A powerful spell is represented by P followed by the rune.
Basic spells are represented by their rune letter.

Example:
Sequence FWFFEA → Encoded as FWPFEA

To prevent “spell spamming,” if a wizard presses the same rune more than twice consecutively, it counts as a single powerful spell.
Example: FFFFW → Encoded as PFW

Additional examples

Input	Output
FEWWFF	FEWPF
FFFFWEEEA	PFPEPA

7. Message Time Reporter
You are building a smart notification system for a popular messaging app.
When someone opens the app, your program needs to tell them exactly how long ago their messages were received, in a way that is easy to read and grammatically correct.
The system should automatically convert the time from seconds into minutes and seconds, or hours, minutes, and seconds, depending on how long ago the message arrived.

Input
Each line contains a single integer N (0 ≤ N ≤ 100,000) — the number of seconds since a message was received. The input ends with -1, which should not be processed.

```
3
1
75
3601
-1
```

Output
For each input N, print a single line:
Message received X ago.

Where X is formatted based on the following rules:
If N < 60: S second(s)
Use “second” if S = 1, otherwise “seconds”.
If 60 ≤ N < 3600: M minute(s) S second(s)
Use correct singular/plural for both minutes and seconds.
If N ≥ 3600: H hour(s) M minute(s) S second(s)
Use correct singular/plural for hours, minutes, and seconds.

```
Message received 3 seconds ago.
Message received 1 second ago.
Message received 1 minute 15 seconds ago.
Message receive 1 hour 0 minutes 1 second ago.
```

8. Optimize the piggy bank

You have a large piggy bank filled with gold, silver, and bronze coins. Each coin type has a value:
1 gold coin = 10 silver coins = 50 bronze coins

1 silver coin = 5 bronze coins

You want to reduce the total number of coins in your piggy bank to make room for more savings, while keeping at least one coin of each type.

Write a program to calculate the minimum number of coins you can have after exchanging coins between types.

Input

- The first line contains a single integer **C** (0 < C < 20) — the number of test cases.
- Each of the next **C** lines contains three integers:
- G S B, representing the number of **gold**, **silver**, and **bronze** coins (0 < G, S, B < 10000).

```
3
500 400 300
5 5 5
59 79 99
```

Output
For each test case, print the **number of gold, silver, and bronze coins** after optimizing the piggy bank.

```
545 9 5
5 5 5
68 8 4
```

9. Alien Signal Translator

An explorer spaceship has intercepted a strange signal from a distant planet.

The signal contains a mix of normal text and numeric codes. Each numeric code represents a character in the alien message using the standard Unicode table.

The mission is to translate the numeric codes into readable characters so the scientists can understand the alien communication.

Input

A single line containing the intercepted message, up to 10,000 characters.

The message may contain letters, numbers, punctuation, and spaces.
Numeric sequences in the message represent Unicode codes (e.g., 65 → A).

```
72 105 32 97 108 105 101 110 33
```

Output

Print the decoded characters corresponding to the numeric codes, separated by a single space.

```
Hi alien!
```

10. The Missing Gloves Mystery

Every winter, the school's Lost & Found box fills up with gloves. Some students manage to lose only one glove from a pair — nobody knows how they disappear, but it keeps happening every year.
You are helping the school caretaker sort through the pile of gloves to find out which ones do not have a matching partner.

A glove is described by three attributes:
Material: Leather, Wool, Fleece, or Cotton
Size: Small, Medium, or Large
Color: Black, Red, Blue, Green, or Brown

A "matching pair" means that two gloves have exactly the same material, size, and color.
There will never be more than one complete pair of the same type in the list — so at most two identical gloves appear.

Task:
Read a list of gloves and identify every glove that does not have a matching partner. If all gloves have partners, print "All gloves accounted for".

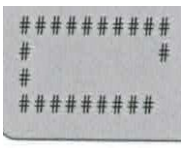
Input
Wool Medium Blue
Leather Small Black
Wool Medium Blue
Cotton Large Red
END END END

Output
Leather Small Black
Cotton Large Red

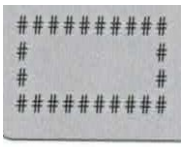
11. Complete The Shape

Your storybook editor asked you to mock up some shape sketches for your soon to be published children's book.
As your book is a preschool shapes book, all you need to do is make sure the rectangles section is complete, since it hasn't been reviewed yet.
Fix any incomplete rectangles.

Input
The input is a (possibly broken) rectangle of'#' symbols, each dimension at least 3 and at most 10.



Output
Complete the given rectangle. If the rectangle is already complete, instead write "Nothing to do".



12. The Colour Code Challenge
The simple browser-based video game, Wordle, took the internet by storm. A player must try to guess a secret word of a given length. After each guess, the game colour codes each letter of the guess according to the correct answer.

- Gray (**-**) means that this letter does not appear in the word at all.
- Yellow (**Y**) means that this letter appears in the secret word, but it's in the wrong spot within the word. (See example 2 for further information).
- Green (**G**) means that this letter appears in the secret word, and it's in exactly the right place.

Write a program which takes the target word and player's guess as input, and outputs the colours **G**, **Y** and **-** according to the above rules.

Input
The first line of input consists of a string of less than 200 characters, the secret word.
The second line of input consists of a string of less than 200 characters, the player's guess.
Both strings are guaranteed to consist of the uppercase letters A to Z. Both strings are guaranteed to be of the same length.

Output
Output the colour coding of the player's guess.

Examples

standard input	standard output
MUSIC MOUSE	G-YY-
BBQERS BUBBLE	G-Y--Y
CRANE CROWD	GG---
SOUND SPOON	G-Y-Y

Note
Explanation of example 1

Letter	Explanation	Colour
M	correct place	G
O	not present in MUSIC	-

U	present, but wrong position	Y
S	present, but wrong position	Y
E	not present in MUSIC	-

Explanation of example 2

Letter	Explanation	Colour
B	correct place	G
U	not present in BBQERS	-
B	present, but wrong position	Y
B	See below	-
L	not present in BBQERS	-
E	present, but wrong position	Y

BBQERS has 2 B's whereas BUBBLE has 3 B's. The first letter in both words match, and so BBQERS has 1 remaining B, and BUBBLE has 2 remaining B's. **As a rule**, since BBQERS has only 1 remaining B, only one of the B's in BUBBLE can be yellow. **As a rule**, we must take the left most remaining B.

13. Say again

Could you repeat that† Could you repeat that? Could you repeat that? Could...
Studies show repetition can help with memory. That's why practice is important. But for us, repetition is best when applied to sentences (lines in a play) or words (spelling tests). It's not as obvious in long strings of lekers or numbers, and that's where a computer is helpful. Can you teach a computer to find a repeating pattern? Maybe another day we'll teach it to remember things...

Input
The input is one line with a repeating pattern of characters (letters and/or numbers, no spaces or punctuation). The pattern starts with the first character and repeats at least 3 times. Upper case and lower case letters are different.
ABBACABBACABBACABB

Output
On the first line of your output, print the missing character(s) to complete the final section of the pattern. On the second line, print the full pattern (just once). Choose the shortest repeating pattern.
AC
ABBAC

Discussion
Remember to choose the shortest pattern. For example, in "AbcAbcAbcAbcAbcAbcAbcA", both "Abc" and "AbcAbc" repeat at least 3 times. But choose the shortest pattern: "Abc".
Reminder: have you run your solution against all of the student data sets?

Additional Example
Input
123451234512345123451234512345123451
Output

2345
12345

14. Spell the Jersey!
For some reason, your school's football team has chosen to spell out the numbers on their jerseys instead of using the usual digits. Being great fans, you're going to be ready to cheer for your favorite players by bringing letter cards so you can spell out their number. Each fan has different favorites, so they each need to bring different sets of letters.

The English spellings for the numbers 0 to 12 are:
ZERO ONE TWO THREE FOUR FIVE SIX SEVEN EIGHT NINE TEN ELEVEN TWELVE

Input
Read a set of integers from 0 to 12, separated by spaces, representing one fan's favorite players. The last integer will be 999, marking the end of the line.
0 1 4 7 9 11 999

Output
Print the same numbers, then a period and a space. Then, in alphabetical order, print all the letters the fan needs to be able to spell any one of the jersey numbers provided.
0 1 4 7 9 11. E E E F I L N N O R S U V Z

Example Input 2
2 3 999

Example Output 2
2 3. E E H O R T W

15. Noise Cancellation Protocol
Cryptography is complicated, expensive, and usually requires someone with a PhD and a very large whiteboard. But what if you don’t need military-grade secrecy? What if you just want to scramble your message just enough so that your nosy little brother can’t read it? Congratulations — you’ve come to the right place.
At the Department of Almost-Secure Communications (DASC), we use a very simple two-phase method called Noise Cancellation Protocol v1.0. Here’s how it works:

Phase 1:
Your program scans the entire message and counts how many times each letter appears.

Phase 2:
Any letter whose total count is N or more (N is the “noise threshold”) is considered “too obvious” and therefore removed entirely from the message. Only the rare letters survive — those are the ones carrying the hidden message.

A human should be able to read the remaining letters, but since humans are unreliable (and dramatic), we also give you the expected number of words and the length of each word. You’re welcome.
One final note: as with all advanced encryption systems, proper data wiping is required. A damp paper towel should be sufficient. Please wipe responsibly.

Input
The message is presented as a grid of letter. The first line of input indicates the number of rows and columns of letters and the noise threshold. The second line begins with the number of words in the solution and then the number of letters in each word. The letter grid follows. Letters are separated by

spaces. There will be no more than 100 letters in the grid.

Example 1
3 15 4
4 2 3 3 9
F L Y W B E G A L K R U B E T
L H G E C K Y U B H L U G A F
K Y F M P U B K F I G O N S Y

Example 2
4 13 6
7 4 1 4 3 5 1 4
P I M L H A K E D P I L A
M E H D E L P R I C A T H
L I G R H P E D L E T I P
H A D G H L O I D O N P D

Output
The program must print the secret message as words on a single line.
WE ARE THE CHAMPIONS
MAKE A MEER CAT GREET A GOON

16. Balanced Energy Matrix
Scientists at the Aurora Research Facility are experimenting with a special energy grid.
The grid is an N×N matrix where each cell contains an energy value. For the grid to operate safely, every row and every column must generate exactly the same total energy — otherwise, the system becomes unstable.

However, some matrices may contain incorrect values due to sensor glitches.
Your job is to analyze the energy matrix and determine:
1. Is the system already stable?
2. If not, can stability be restored by adjusting *exactly one cell*?
3. Or is the system too corrupted to fix with a single correction?

Input
The first line contains an integer N ($3 \le N \le 100$).
Then N lines follow, each containing N space-separated integers — the current energy readings.
Every value has an absolute value no greater than 10 000.

Output
Your program must print one of the following:
1. STABLE - If all rows and columns already sum to the same value.
2. CAN FIX - If changing one and only one cell can make the matrix stable.
In this case, print a second line:
row column new_value
If multiple fixes exist, you may output any valid one.
3. UNFIXABLE - If no single change can make all row and column sums equal.

Example

Input	Output
3 2 3 6	UNFIXABLE

9 5 1
1 3 9

3	CAN FIX
1 1 1	2 3 1
1 1 2	
1 1 1	
4	STABLE
5 14 15 1	
9 8 6 12	
5 11 11 8	
16 2 3 14	

17. Nostromo's Armstrong Number Detector

In the year 2122, the commercial towing spaceship Nostromo is on its return trip to Earth. The ship's onboard computer, MU-TH-UR 6000, has detected an unknown signal from a nearby planetoid. The crew is awakened from stasis to investigate. As they prepare to land on the planetoid, MU-TH-UR runs a diagnostic to ensure all systems are operational. One of the checks involves validating the integrity of the ship's power cells, which are labeled with unique numerical codes.

Your task is to help MU-TH-UR create an Armstrong Number Detector. An Armstrong number is a number that is the sum of its own digits each raised to the power of the number of digits. For example, 153 is an Armstrong number because (13 + 53 + 33 = 153).

Input
A single positive integer number.
371

Output
Return True if n is an Armstrong number, or False otherwise
True

18. SOS Morse

Imagine you are stranded on a deserted island with access to rudimentary technology for sending messages in Morse code. You cannot transmit a voice or text message, but you can send Morse sequences using a device that emits and receives intermittent signals. In an emergency, you want to send the universal distress signal, SOS (…---…), in Morse code to increase your chances of being rescued. Sending the pattern …---… at least three times will allow a nearby emergency boat to detect your presence. Sending SOSOS (…---…---…) will count as two times sent, saving the transmission of an S.

Input
The input consists of a line with dots and dashes.

Output
The output will print "Saved" if the SOS Morse pattern appears 3 times or more in the message. Otherwise, the output will be "Not saved".

Example
Input
`...---...-.--.--....---...---...`

Output
```
Saved
```

Input
```
..--...-.--.---....---...---...
```

Output
```
Not saved
```

19. Pattern
Starting with 2 and using a common difference of 3, we can get the sequence 2, 5, 8, 11, 14, …

Starting with 2 and using common differences of 3 and -2, we can get the sequence 2, 5, 3, 6, 4, 7, … (we add 3, subtract 2, add 3 again, subtract 2, and so on).

Given the first few terms of a sequence, the program you must write should determine the smallest group of common differences, in order, that will generate the sequence if you are given the first term.

The first number in the input is the number of terms, ��. This is followed by �� numbers, the terms of the sequence, separated by spaces. The program should output the group of integers you have found separated by spaces.

Examples:

`Input: 5 2 5 8 11 14`	`Output: 3`
`Input: 6 2 5 3 6 4 7`	`Output: 3 -2`
`Input: 12 3 1 2 5 3 4 7 5 6 9 7 8`	`Output: -2 1 3`

20. Level Reset Protocol
The Chrono Tower is a mysterious structure where explorers attempt to climb through its endless levels. However, certain levels are protected by ancient time seals.
You begin your journey at Floor 1. On each turn, you try to advance to the next floor.
Some floors are Time-Sealed Floors — these are the floors whose numbers are perfect squares (1, 4, 9, 16, ...).
The first time you reach a time-sealed floor, the seal activates, you are expelled, and you must restart at Floor 1.
The second time you reach that same sealed floor, the seal recognizes you and lets you pass.
This cycle continues for every sealed floor in the tower.

Example
During the first 15 turns, you would reach the following floors:
1, 1, 2, 3, 4, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1

Task
Given a turn number T, determine which floor you will be on at turn T.

Input	Output
4	3
10	5
18	4

Hard Challenges

1. Roman Numerals

The ancient Romans used a numeric system different to the Arabic numeral system we use today. In the Roman system, numbers are represented by combinations of the letters 'I', 'V', 'X', 'L', 'C', 'D', and 'M'. Each letter has a fixed positive integer value, as shown below:

Symbol	I	V	X	L	C	D	M
Value	1	5	10	50	100	500	1 000

To represent a given positive integer N in Roman numerals, for the most part, one simply writes down the highest Roman numeral symbol with value S which is less than or equal to N, then recursively writes the Roman notation for N − S.

For example, to represent the number 738 as a Roman numeral, one finds the highest valued symbol in the table above which has value at most 738. In this case, it's D with a value of 500. What now remains is to represent 738 − 500 = 238. Now we take C with a value of 100 twice, leaving 238 − 2 · 100 = 38. We then take X three times, V once, and finally I three times. This results in a Roman representation of 738 as DCCXXXVIII.

However, the Romans also made use of subtractive notation in their numeral system to reduce the total number of symbols used. This means that numbers like 4 and 9 were represented as IV and IX respectively. Similarly, 40 and 90 were represented as XL and XC respectively, as well as 400 and 900 being represented as CD and CM respectively.

Therefore, the numbers from 1 to 10 are represented as

Arabic number:	1	2	3	4	5	6	7	8	9	10
Roman number:	I	II	III	IV	V	VI	VII	VIII	IX	X

noting that 4 and 9 are the only digits requiring subtractive notation. Similarly, multiples of 10 from 10 to 100 are represented as

Arabic number:	10	20	30	40	50	60	70	80	90	100
Roman number:	X	XX	XXX	XL	L	LX	LXX	LXXX	XC	C

Your task is, given two numbers in the Roman numeral system, output their sum as a Roman numeral.

Input
First line of input contains a single string denoting the value of a positive integer A ($1 \le A \le 3\,999$) written as a Roman numeral.
Second line of input contains a single string denoting the value of a positive integer B ($1 \le B \le 3\,999$) written as a Roman numeral.

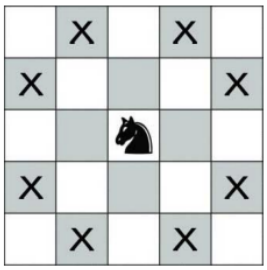
Output
Output a single string, denoting the Roman numeral value of A + B.

Examples

Input	Output
II III	V
IV XI	XV
XLIX L	XCIX

2. Knight Moves
Given a chessboard with N rows and M columns, a start position A and an end position B, the program you must write must calculate the fewest number of moves it takes a knight* to travel from A to B. In all the test cases, it is always possible for the knight to travel from A to B.

The first input line contains N and M. The second line contains two integers, the row and column, respectively, of square A. The third line contains two integers, the row and column, respectively, of square B. Note that the rows are numbered 1 to N, and the columns from 1 to M. All input values are separated by a space.



*In one move, a knight can move two squares vertically and one square horizontally, or two squares horizontally and one square vertically. The image shows all the legal moves that the knight can make.

Examples:

```
Input:          Output: 2
8 8
4 4
4 2
Input:          Output: 3
```

```
3 3
1 1
2 1
Input:          Output: 4
3 6
2 1
1 6
```

3. WORD SQUARE

Write a program that will find given words in the 8 by 8 word square below. You may assume that the words appear in rows and read forward (L to R) or backwards (R to L). [Hint: As you use the same word square all the time you can save time by hardcoding it into your program.]

T	H	E	Q	T	T	E	P
L	R	E	K	A	W	R	E
T	S	E	T	N	O	C	I
X	Y	C	O	L	T	Z	J
R	C	H	I	L	E	S	Q
U	M	N	X	B	E	S	T
Y	X	K	O	O	C	Q	F
G	H	I	K	C	O	R	B

Your program should print out the word and indicate the row and column where the word begins and in which direction it proceeds (forwards or backwards).

Input: COLT

Output: COLT, row 4, column 3, forwards

Test your program with

(a) THE

(b) BEST

(c) CONTEST

4. Toll Roads

Six cities, A, B, C, D, E and F are connected by toll roads. Some roads can be zero toll roads. Write a program to work out the cheapest route to visit each city once only, starting at A and ending at A, and give the total toll to be paid. If there is more than one cheapest route, you may provide any one of them as answer. You may travel a road in any direction. The tolls are given in the following order:
A-B; A-C; A-D; A-E; A-F; B-C; B-D; B-E; B-F; C-D; C-E; C-F; D-E; D-F; E-F

Example 1:
Input: (in Rand) 6, 3, 2, 3, 10, 1, 6, 7, 4, 5, 1, 5, 10, 7, 3
Output: A D C B F E A, 18

Example 2:
Input: (in Rand) 10, 15, 7, 10, 7, 7, 9, 6, 7, 11, 12, 8, 5, 14, 9
Output: A D E B C F A, 40

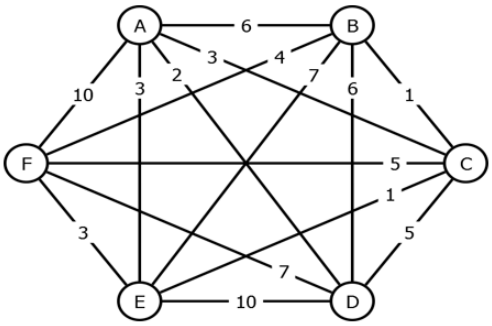
Record the results of the following test cases:

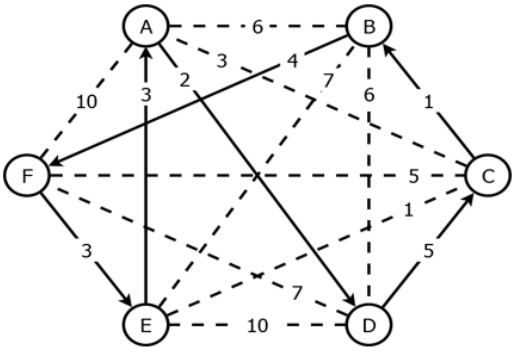
a: 9, 8, 7, 6, 5, 4, 3, 2, 1, 2, 3, 4, 5, 6, 7

b: 6, 0, 7, 0, 8, 0, 1, 2, 3, 0, 5, 6, 7, 8, 9

c: 12, 10, 14, 4, 11, 7, 2, 2, 10, 3, 6, 9, 8, 10, 1

Illustration of Example 1:
(Please note: you do NOT have to produce an illustration.)





5. Phone password guessing

Your friend has bet you $20 that you cannot figure out their 4-digit phone passcode given 3 guesses. Using the smudges left on their screen over the numbers, and your knowledge of your friend's passcode rules, try to figure out their passcode.

Your friend always chooses a passcode where the first 2 digits (as a 2-digit number) are the square of the second 2 digits (as a 2-digit number).

Input
The dataset represents a phone passcode input screen:
123
456
789
X0X

The dataset will be marked with C or U. C = clearly a number in the passcode.
U = unclear, possibly used in the passcode.
If a location is a digit or an X, then it isn't in the passcode.

EXAMPLE INPUT:
1C3
4U6
789
XCX
For this input, 2 and 0 are definitely used. 5 might be. No other digits are in the passcode.

Output
If you only find one possible combination of digits which could be the passcode, output: CODE IS: ####
If two or three possible passcodes are found, output each on a new line as: POSSIBLE MATCH: ####
POSSIBLE MATCH: ####
POSSIBLE MATCH: #### (if 3 codes are found, output 3 lines)
If four or more possible passcodes are found, output: WILL NOT WIN BET
If no possible passcodes exist, output: COULD NOT DETERMINE CODE

EXAMPLE OUTPUT:
CODE IS: 2505

6. We have got you covered

This year we're experimenting with the placement of WiFi hotspots ... but we're not sure we've got it right.

The layout for each contest site is represented by an L by W grid with student locations marked with a **#** character.

WiFi hotspot locations are provided as X, Y coordinates that correspond with the length and width of the room grid, respectively. WiFi hotspots can be placed outside of the contest room. WiFi hotspots can also occupy the same location as a student.

Each WiFi hotspot has a an associated signal radius inside which reliable network coverage can be guaranteed. Students must be strictly inside a hotspot's radius to get reliable coverage.

Student and hotspot locations will be at the center of their grid square.
Given a set of contest sites, student locations, and WiFi placements, tell how many students will get reliable network coverage at each site.

Input

Input will begin with an integer S (0 < S <= 20) representing the number of sites.
The description for each site will begin with a single integer H (0 < H <= 20) representing the number of wireless hotspots at that site.
The following H lines will describe a single hotspot. A hotspot description begins with a pair of integers X and Y (-50 <= X, Y <= 250) representing its coordinates and ends with an integer R (0 < R <= 300) representing the radius of the hotspot's signal. The X coordinate corresponds with the length of the room and the Y coordinate with the width.
Next come a pair of integers L and W (2 <= L, W <= 200) representing the length and width of the room at a contest site. The first character of the first line is at coordinates 0, 0.
The following L lines will contain a string W characters long consisting of either a **.** or a **#**. A **.** represents an empty space and a **#**
represents a student location.

```
2
3
0 1 2
3 3 1
5 -2 4
4 4
#.#.
.##.
.##.
#..#
2
1 4 1
2 2 1
3 5
.....
....#
.#...
```

Output

For each contest site, output a single line that tells how many students get coverage with that site's WiFi configuration.
For sites where a single student gets coverage, use the following format: "Site #: 1 student gets coverage" For other sites, use the following format: "Site #: # students get coverage"

```
Site 1: 6 students get coverage
Site 2: 1 student gets coverage
```

7. Ciphers

In this problem, each letter of the alphabet corresponds to a number using the scheme: a=1, b=2, c=3, ... y=25, z=26. To encode a message, an encryption key word is added to the message. The key word is the first word in the message that is five or more characters long. For example, if the message were: "give me liberty or give me death," the key word would be "liberty." The encrypted message would be: s r x j e y k u k g w l s n d p k a w g d p n c y z

The first letter, s, results from adding 7 for the letter "g" to 12 for the letter "l." The sum is 19, which creates the letter "s." The other letters are calculated in a like manner. The algorithm you develop should wrap around to the beginning of the alphabet if the sum of two letters exceeds 26. For example, when the letter "l" (12) is added to "y" (25), the sum is 37. Since there is no 37th letter in the alphabet, you should subtract 37 from 26 to get the number 11, which equates to the letter "k." The key word is repeated indefinitely to the end of the message.

give me liberty or give me death
libe rt ylibert yl iber ty liber
---
srxj ey kukgwls nd pkaw gd pncyz

(Spaces are left between the words in this example for clarity. Do not include spaces in your results.)
Every message will have at least one five-letter word.
Spaces are not encrypted and not included in the output. Only messages in lowercase letters will be entered. No punctuation or special characters will be entered.

Input
Consist of a sentence or phrase entered via the standard input device and the key.
a stitch in time saves nine
stitch
Output
t m c c w k a c w n l u x m j p h a g c w y

Additional examples
Input
it is a rainy day
rainy
Output
a u r g z j b r b x v b h

Input
it only hurts when i laugh
hurts
Output
q o g h e g c m l m a r z y g q g s o z p

8. RoboRumble

RoboRumble is an exciting game where robots of different models fight each other in an arena. Each robot has unique specifications:
Name: The robot's model name (no spaces).
Category: Can be Mech, Drone, Cyborg, AI, Nano, or Quantum. A robot may have a vulnerability or

armor against attacks from certain categories.
Vulnerability: If the robot is attacked by a robot of this category, the attack's damage is doubled.
Armor: If the robot is attacked by a robot of this category, the attack's damage is halved.
Integrity: A positive integer (up to 250) representing the robot's durability. When it drops to 0 or below, the robot is destroyed.
Abilities: Each robot has two abilities. Each ability has:
A name (no spaces)
An integer energy cost (1–5) to use the ability
An integer attack power (up to 200) indicating damage

At the start of the fight, robots have full integrity and zero energy. Robots take turns: each turn, the robot gains 1 energy and can use one ability if it has enough energy. If it can use both, it chooses the one with the higher attack power. The fight continues until one robot's integrity drops to 0 or below.
Write a program to simulate battles between various RoboRumble robots.

Input
The first line contains the number of robots.
Next lines contain robot details: Name Category Vulnerability Armor Integrity Ability1_Name EnergyCost Damage Ability2_Name EnergyCost Damage.
If a robot has no vulnerability or armor, it is indicated by X.
Then, a line with the number of battles to simulate.
Each battle is described by a line with the names of the two robots, in order of taking turns.

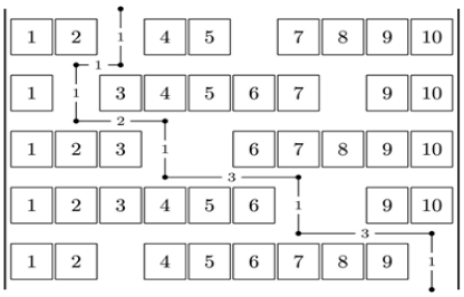
Example Input:
4
RoboX Mech Drone Nano 150 LaserBlast 2 50 RocketPunch 4 100
SkyDrone Drone Mech X 120 SonicBoom 1 30 AeroStrike 3 80
NanoCyborg Nano AI Quantum 100 HackPulse 1 40 NanoSlash 3 90
QuantumAI AI X Mech 80 MindBlast 2 60 QuantumBeam 4 120
3
RoboX SkyDrone
NanoCyborg RoboX
QuantumAI NanoCyborg

Output
For each battle, print:
<Winner> defeats <Loser> with <Remaining Integrity> integrity remaining.
RoboX defeats SkyDrone with 40 integrity remaining.
NanoCyborg defeats RoboX with 20 integrity remaining.
QuantumAI defeats NanoCyborg with 10 integrity remaining.

9. SHORTEST PATH THROUGH THE HALL

The hall is packed wall to wall with rows of chairs, but in each row there are exactly two chairs missing. The chairs in each row have numbers from 1 to 100. Write a program that will calculate the length of the shortest path from the front to the back of the hall.

Each chair is 1 unit wide and each row is 1 unit deep (from the front of a chair to the front of the chair behind it). It is not possible to move diagonally. You may start in front of any gap in the front row and end behind any gap in the last row. You always walk through the middle of a gap. Illustrated is the shortest path through a hall, with five rows of chairs. In the illustration the hall is only 10 chairs wide instead of 100.

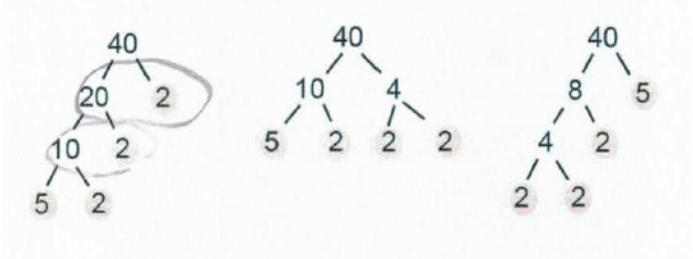


The first number in the input will contain the number n – the number of rows. The next n lines will have two numbers, separated by a space, indicating where the gaps are.
Example Input:
5
3 6
2 8
4 5
7 8
3 10

Output: 14

Use the following sets of data to test your program
a:
4
5 8
9 3
1 5
2 7
b:
6
14 84
15 88
95 96
17 42
15 62
10 100

10. Factored
How do trees get online? They just log in.
Now that you are online, let's look up what a factor tree is. A factor tree is a special diagram where we find the factors of a number, then the factors of those numbers, etc, until we can't factor any more. In other words, a factor tree shows the number that come together to create a large number.



Input

You will receive a list of integers (32-bit). Zero on a line by itself ends input

```
40
9
18
7
25
13
12
0
```

Output

If the integer can be factored into the multiplication of smaller integers (not including 1 and itself), list all of the possible factors as follows:

```
40=(20*2), (10*4), (10*2*2), (8*5), (5*4*2), (5*2*2*2)
9=(3*3)
18=(9*2), (6*3), (3*3*2)
25=(5*5)
12=(6*2), (4*2), (3*2*2)
```

Any factors which can be broken down further should be listed as separate factor pairs. When listing factors, always list the largest factor groups first. Separate factor groups with a comma. Inside factor groups, always show the multiplication as: (largestNumber * smallerNumber [* smallerNumber..]). It will be counted as wrong to list the factor of 12 as: (2*6),(3*4),(2*2*3). List the factor pairs in descending order. It will be counted as wrong to list the factor of 18 as: (3*3*2),(9*2)