codescrim

Codecember 2017 Problems

Problems featured in Codecember 2017. Problems were original creations by: Emily Peterson, Matthew Zhang, Saif Ahmad, Ajai Shankar, Rohit Mohan, and George Thayamkery. Or were taken/modified from sources that released problems under The Creative Commons Attribution License

For every problem use `System.in` or `stdin` to take in test cases

To emulate this on your own enviroment you can use `<` operator in a terminal to pipe a text file into a program

```
$ java SomeClass < TextCases.txt
```

Codecember

Math is Cool

Worth 10 point(s) - Runtime Limit: 5 seconds

Introduction

Sandra and Mortimer are computer science students, and they think that math is cool. So cool, in fact, that like many people you hear about in contest problems, they love to play math games. In their favorite game called “First to One”, the two friends first pick a number N, and then compete to see who can change N into 1 within the least number of moves. A “move” can consist of one of three options:

1. Subtracting 1 from N
2. If N is divisible by 2 (no remainder), dividing N by 2
3. If N is divisible by 3 (no remainder), dividing N by 3

Now write a program which will output the minimum number of moves required to go from N to one. Then you’ll be able to join in the merriment and destroy Mortimer and Sandra at their own game!

Input

On the first line of input will be an integer T denoting the number of test cases to follow. Next will be T lines, each containing a single integer N (0 < N < 1,000,000).

```
4
2
17
52
14236
```

Output

For each test case, print a line containing the least number of moves it takes to go from N to 1 using the above options.

```
1
5
6
15
```

Ascending Order

Worth 9 point(s) - Runtime Limit: 3 seconds

Introduction

Sarah likes to play a game where given a number, she tries to find the last number before it where the digits in the number are in ascending order. Numbers like 8, 123, 555, and 224488 are examples of numbers with ascending order digits. Numbers that do not have this property, like 20, 321, 495 and 999990, do not contain ascending order digits.

Write a program that finds the largest number with ascending order digits $\leq N$.

Input

The first line of the input gives the number of test cases, T. T lines follow. Each line describes a test case with a single integer N where $0 \leq N \leq 10^{18}$.

```
4
132
1000
7
111111111111111110
```

Output

For each test case, output one line containing the largest number with ascending order digits.

```
129
999
7
99999999999999999
```

Biologist

Worth 8 point(s) - Runtime Limit: 1 seconds

Introduction

A biologist is growing different populations of bacteria in a common petri dish in order to see which species survive better in the competitive environment. Though she can estimate the size of the different bacteria populations when squinting through her microscope, she would much rather write a program to analyze a high-quality photograph of the petri dish to more accurately observe the growth of each species. Help the biologist write her program!

Input

On the first line of input will be an integer T denoting the number of test cases to follow. For each test case there will then be a line containing two space-separated integers R and C. The next R lines will contain C characters each, comprising the petri dish photograph to be analyzed. Each character will be a capital letter representing a unit of pixels that can be recognized as a certain species of bacteria present in the dish.

```
1
6 8
COMPSCII
SSMPSSSI
SSMMSPPP
IOSMPCOP
CISPPIOQ
CIMPOSSS
```

Output

For each test case photograph, your program should count up the number of “colonies” per species of bacteria as well as the number of pixel units (characters) for each bacteria species. A colony is a comprised of a group of adjacently connected pixel units of the same species (not including diagonal connection). For each species of bacteria observed in the photograph, print a line beginning with the character representation of the species followed by the number of detected colonies of that species and then the total number of pixel units recognized of that variety. These output lines should be sorted in order of the total number of pixel units recognized per species in descending order (the most prevalent species will be outputted first). Ties for this ordering will be broken alphabetically by the character representation of the species. Lastly, in your output, follow each test case with a blank line divider.

```
S 4 14
P 3 10
I 4 7
M 2 6
C 4 5
O 4 5
Q 1 1
```

Good luck!

Compiler For a Day

Worth 8 point(s) - Runtime Limit: 1 seconds

Introduction

Ever wish that you could put yourself in the compiler's shoes and see what it's like to decipher code? I know that you have. But the compiler's job entails more complications than you might expect. The Java compiler overlooks white space and similar formatting when interpreting code, except of course when spaces separate key words and names and such. This comes into play for something as simple as declaring and initializing a variable. Write a program to validate whether or not a line of code which declares and initializes an integer variable fits Java syntax and is free of any compile-time errors.

The first line of input will contain an integer T denoting the number of test cases to follow. The next T lines will each contain a line of code to be interpreted.
For each line of code, print out `VALID` if the line is free of compile-time errors or print `INVALID` if that is not true and the line is syntactically incorrect for Java, assuming that the line of code is contained within the main method of a defined class and there are no other variables that have been declared. You need not worry about the integer literal being out of range (overflow).
A valid line will contain zero or more spaces, followed by the key word "int", followed by one or more spaces, followed by the space-less variable name (which may contain letters or numbers), followed by zero or more spaces, followed by an equal sign, followed by zero or more spaces, followed by a literal integer number, followed by zero or more spaces, followed by a semicolon, followed by zero or more spaces after that.For example:

```
int jimbo = 15;
```

Input

The first line of input will contain an integer T denoting the number of test cases to follow. The next T lines will each contain a line of code to be interpreted.

Sample Input

```
10
int bob = 5;
INT bill = 10;
intcaroline = 3;
int   su5an=222  ;
int JO3Y= 20
  int q = -30;
int num = 4.33;
int num = “45”;
int blah = otherVar;
int nice one = 300;
```

Output

For each line of code, print out `VALID` if the line is free of compile-time errors or print `INVALID` if that is not true and the line is syntactically incorrect for Java, assuming that the line of code is contained within the main method of a defined class and there are no other variables that have been declared. You need not worry about the integer literal being out of range (overflow).

A valid line will contain zero or more spaces, followed by the key word “int”, followed by one or more spaces, followed by the space-less variable name (which may contain letters or numbers), followed by zero or more spaces, followed by an equal sign, followed by zero or more spaces, followed by a literal integer number, followed by zero or more spaces, followed by a semicolon, followed by zero or more spaces after that. For example:

```
int jimbo = 15;
```

Sample Output

```
VALID
INVALID
INVALID
VALID
INVALID
VALID
INVALID
INVALID
INVALID
INVALID
```

Location Awareness

Worth 8 point(s) - Runtime Limit: 1 seconds

Introduction

Many modern computers have Location Awareness features. Phones, tracking systems, and self-driving vehicles have the ability to determine their current location. Often this is accomplished using a trilateration algorithm. If the device can receive signals from three sources whose locations are known, then it can determine its location from that data.

For this problem, you will write a trilateration algorithm (explained below) for an autonomous robot using signals from three towers positioned around a square arena. The arena is an integer grid with walls at the four lines defined by $x = 100$, $y = 100$, $x = -100$, and $y = -100$. The robot may be positioned anywhere within the arena. Tower 1 is located at (x, y) position $(0, 100)$, tower 2 at $(-100, -100)$, and tower 3 at $100, -100$.

Here's how the system works: the tower broadcasts distinct signals that the robot can receive. The towers are all powered by one common battery. When the robot is near a tower, its signal strength is high, but the farther the robot is from the tower, the weaker the signal. The strength of a tower's signal is given by the following equation:

$$s = P/d^2$$

The variable P is the transmission power and d is the distance from the tower to the robot. When the battery is fully charged the signal is very strong. But over time, as the battery's energy is used, the signal power is reduced. So P is the same for each tower, but it changes over time. Also, the robot has no direct way to measure P. So it is not possible to make an exact calculation for the distance to a tower using the signal strength. You'll have to think of how to use all three signals to solve this problem.

Use the signals from the three towers to determine the robot's location on the grid.

Sample Input

Each line of input has three floating-point number separated by one or more spaces. These numbers are the signal strengths from towers 1, 2, and 3, in that order, for each location of the robot. The input ends with three zeros.

```
5.432 2.716 2.716
6.733 0.956 1.284
501.345 2.102 1.878
2.207 2.644 662.852
0 0 0
```

Sample Output

For each input line, print the exact integer x and y location of the robot.

```
0 0
21 35
-14 99
93 -90
```

Fast Fibonacci

Worth 6 point(s) - Runtime Limit: 2 seconds

Introduction

The Fibonacci Sequence is really well known. In fact, it even has it's own day on November 23rd.
Unfortunately as the sequence goes up higher and higher, the time it takes to solve increases exponentially. That is if you don't do any tricks!

Write a program that can quickly determine the nth Fibonacci number in the sequence.

Input

The first line of input is T, the number of test cases. Each test case consists of one line containing n, the term of the sequence to generate, where $1 \leq n \leq 100$.

```
3
4
10
20
```

Output

Output the nth term in the Fibonacci Sequence.

```
3
55
6765
```

Structural Integrity

Worth 6 point(s) - Runtime Limit: 1 seconds

Introduction

Billy and his friends are building snow forts. Billy, though, always having to come out on top, builds his out of ice. Since ice is a lot heavier than snow and harder to use, Billy enlists his engineering skills to determine the weak points in his fort. And because he is as creative as he is interesting, his fort is in the shape of a cube. To "help" Billy (aka do his work for him), we'll divide his fort into 1x1x1 blocks of ice, each with coordinates of (x, y, z). Each block is also given a number to describe its strength, but since Billy isn't the greatest builder, all of these values are initially 0. Billy will also ask us to do a few things while building. When Billy wants us to change the strength of a certain ice block, he will ask the following:

```
UPDATE x y z W
```

which means that we should change the strength of the block at (x, y, z) to W.

He will also want to know about the overall integrity of his fort, for which he will ask:

```
QUERY x1 y1 z1 x2 y2 z2
```

which means that we should tell him the sum of the strengths of blocks whose x coordinate is between x1 and x2 (inclusive), y coordinate between y1 and y2 (inclusive) and z coordinate between z1 and z2 (inclusive).

Input Format

The first line contains an integer T, the number of test-cases.
For each test case, the first line will contain two integers N and M separated by a single space.

N defines the N * N * N matrix.
M defines the number of operations

The next M lines will contain either

```
1. UPDATE x y z W
2. QUERY x1 y1 z1 x2 y2 z2
```

Sample Input

Constraints

- $1 \leq T \leq 50$
- $1 \leq N \leq 100$
- $1 \leq M \leq 1000$
- $1 \leq x1 \leq x2 \leq N$
- $1 \leq y1 \leq y2 \leq N$
- $1 \leq z1 \leq z2 \leq N$
- $1 \leq x, y, z \leq N$
- $-10^9 \leq W \leq 10^9$

```
2
4 5
UPDATE 2 2 2 4
QUERY 1 1 1 3 3 3
UPDATE 1 1 1 23
QUERY 2 2 2 4 4 4
QUERY 1 1 1 3 3 3
2 4
UPDATE 2 2 2 1
QUERY 1 1 1 1 1 1
QUERY 1 1 1 2 2 2
QUERY 2 2 2 2 2 2
```

Sample Output

```
4
4
27
0
1
1
```

Pascal's Snowforts

Worth 6 point(s) - Runtime Limit: 1 seconds

Introduction

In Antarctica, there are plans to create a new triangular snow fort with various items in it. In order to achieve this, the lead architect suggested using Pascal's Triangle in order to create the fort and decide what goes in it.

Here is how Pascal's Triangle Works:

1 - Start out the triangle of numbers like this:

```
 1
1 1
```

2 - Every row has one more number then the last. For each number, take the two numbers above it and add them together and make that the value.

```
   1
  1 1
 1 2 1   // 2 comes from the two numbers above it added together
1 3 3 1  // 3 comes from the two numbers above it added together
```

Write a program for the architect that generates blueprints of the fort given a certain size.

Input

The input starts with one line containing I, the number of test cases. Each test case contains one line with a number that gives n, the size of the fort ($1 \leq n \leq 40$).

```
1
4
```

Output

The blueprint of the fort up to n with the correct spacing.

1

1 1

1 2 1

1 3 3 1

Skyline

Worth 5 point(s) - Runtime Limit: 1 seconds

Introduction

There are few things as charming as the serene view of skyscrapers forming a city skyline in the wintertime. Write a program to display one such skyline!

Input

The first line of input will contain an integer T denoting the number of test cases to follow. For each test case, there will be a single line that will begin with an integer S, the number of skyscrapers in the skyline. Then there will be S space separated integers to follow, each of these giving the story height of its respective skyscraper in order from left to right.

```
2
4 3 6 5 2
5 7 4 2 3 5
```

Output

For each test case, output an ascii skyline as depicted below. Separate each new skyline with a blank line.

```
    _
   |#| _
   |#||#|
 _ |#||#|
|#||#||#| _
|#||#||#||#|
|#||#||#||#|

 _
|#|
|#|          _
|#| _       |#|
|#||#|    _ |#|
|#||#| _ |#||#|
|#||#||#||#||#|
|#||#||#||#||#|
```

Letter Distribution

Worth 5 point(s) - Runtime Limit: 1 seconds

Introduction

The use of letters in the English language is not evenly distributed. For example, the letters E and T are used far more often than the letters X and J. Make a histogram of the frequencies of different letters in a given paragraph of text.

Input

```
We the People of the United States, in Order to form a more perfect Union, establish Justi
ce, insure domestic Tranquility, provide for the common defence, promote the general Welfa
re, and secure the Blessings of Liberty to ourselves and our Posterity, do ordain and esta
blish this Constitution for the United States of America.
```

Output

The program must count the number of occurrences of each letter of the input and sort the letters by popularity, from most popular to least. Upper case and lower case letters are considered the same for counting purposes. Spaces and punctuation are to be ignored. Two or more letters of equal popularity must be sorted alphabetically. the program must print a horizontal histogram of the sorted letter counts as shown below so that one "*" is displayed for each occurrence of a letter.

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

Snow Forts

Worth 5 point(s) - Runtime Limit: 2 seconds

Introduction

Billy and his friends are making snow forts on a coordinate grid. Each of his friends built their fort on a random point on the grid, and Billy, being the narcissist he is, built his at (0, 0). As a master strategist, Billy wants to know which of his friend's forts he should attack first. Billy has a limited number of snowballs, though, so he can't attack *every* fort. With the snowballs he has, he can only attack K forts. If Billy has N friends, find the K closest forts for Billy to attack.

Input

Input consists of I test cases, where each test case will have one line with two numbers N and K, followed by N lines of Billy's friends and their location on the grid.

```
4 4
Tom -4 48
Joe -19 8
Karl 0 -18
Marx 37 -44
5 3
Beth 15 15
Melvin 29 49
Timothy -18 30
William 45 35
Reginald 39 -10
```

Output

For each set, output the order in which Billy should target his friends, from the closest fort to the farthest fort.

```
Karl
Joe
Tom
Marx
Beth
Timothy
Reginald
```

Barcode Reader

Worth 5 point(s) - Runtime Limit: 1 seconds

Introduction

Imagine a world without barcodes. Checkout lines at grocery stores would take years to get through, and inventory checking would be a nightmare. Well, let us not dwell too long on those dark thoughts. Rather, write a program that "scans" a barcode and looks up the item the barcode is attached to.

Input

The first line of input will contain an integer Z, the number of items catalogued as products in your store. The next Z lines will each contain a 10 digit barcode where each digit is either a "1" or a "0" (1 representing black ink), followed by a space and then the name of that item. After that, there will be an integer T denoting the number of barcodes to be scanned. The next T lines will each contain a 10 digit barcode.

```
5
1001110101 White Toaster
0010110010 Green Beans
1011101000 USB
0001110101 Detergent
1101110110 Nuclear Reactor
5
1011101000
0010110010
1101110110
0010110010
1110000000
```

Output

For each barcode you scan, print out the name of that item. If the barcode is not listed in your catalogue, print "UNRECOGNIZED"

```
USB
Green Beans
Nuclear Reactor
Green Beans
UNRECOGNIZED
```

Iditarod

Worth 4 point(s) - Runtime Limit: 1 seconds

Introduction

The Iditarod is an annual sled dog race 350 miles in length and taking about 15 days to complete. The best sled dog mushers in the world gather in Alaska to compete in the toughest race conditions, facing blizzards, sub-zero temperatures, and powerful winds. This year though, the race will need to pick a new course (due to blockages). Help calculate the distance of the new Iditarod course! To determine the race path, a sled dog team will be sent out into the Alaskan wilderness following its musher's commands. Given the commands of the musher as she navigates the terrain for a new course, determine the distance between the starting and finishing points.

Input

There will be several lines of input. Each line will contain a sled dog command (one of four) from the musher. If the command is "MUSH", the sled dogs will advance forward in their current direction for 10 miles. If the command is "GEE", the dogs will turn 90 degrees to the right, and if the command is "HAW", the dogs will turn 90 degrees to the left. Input will be terminated by the command of "WOAH".

```
MUSH
MUSH
GEE
MUSH
MUSH
HAW
MUSH
HAW
HAW
MUSH
GEE
MUSH
GEE
MUSH
WOAH
```

Output

Print out the distance between the starting point of the sled dog team and finishing point. The distance should be rounded into an integer.

Number Conversions

Worth 4 point(s) - Runtime Limit: 3 seconds

Introduction

Turns out there are more than one way to count numbers.

Write a program that accepts Roman numeral numbers and outputs the value in Arabic Numerals. Your program must handle the Roman numeral values `M`, `D`, `C`, `L`, `X`, `V` and `I`, in both upper and lower case. For example: `CMI` is `901` and `MCMXCI` is `1991`.

Input

The first line contains N, the number of test cases. Each test case consists of one line containing a string of roman numeral characters.

```
3
XII
XXXIX
DCXCIX
```

Output

Print out the Arabic Numeral representation of each roman numeral test case.

```
12
39
699
```

Sorted Odds

Worth 4 point(s) - Runtime Limit: 1 seconds

Introduction

Given a list of integers, sort the odd numbers while leaving the even numbers in place.

Sample Input

Each test case consists of a line of space separated integers. Assume that none of the numbers will be negative and nonzero.

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

Sample Output

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

Santa's Guessing Game

Worth 4 point(s) - Runtime Limit: 5 seconds

Overview

So you ended up on Santa's naughty list. Well luckily for you Santa has a way to get back on the nice list. All you have to do is guess what number he is thinking of from 0 to n. To make it more fair, you can guess a number, and he will tell you if it's higher or lower than the number he is thinking of. But you only have $\log_2 n$ guesses before you're out of luck.

Can you outsmart Santa?

Write a program that can guess a number from 0 to n in $\log_2 n$ attempts.

Input and Output

The format for this problem is different than the others. Your program will be talking to another program (representing Santa). You can ask Santa if the number he is thinking of is something, and he will respond with either `LOWER` if his number is lower than yours, `HIGHER` is his number is higher than yours, and `CORRECT` if you guessed the number right.

Here is how it will work:
Santa will tell you n where $0 \leq n \leq 100,000,000$.

```
10
```

Then you can make your guess of what number he is thinking of

```
3
```

Then Santa will respond with either `HIGHER` , `LOWER` , or `CORRECT`

```
HIGHER
```

Then you can make another guess

```
7
```

...and so on and so forth

If you can guess Santa's number in the correct amount of attempts, you can get back on the nice list (and solve this problem).

NOTE: Do not use `kb.hasNext()` in your input loop, instead use `kb.hasNextLine()`

Sample Input

```
10
HIGHER
LOWER
CORRECT
```

Sample Output

```
3
7
5
```

Student Grading

Worth 4 point(s) - Runtime Limit: 3 seconds

Introduction

It is the almost the end of the first semester and your teacher has to make a student report. Can you write a computer program to help your teacher out?

Write a program that outputs each students, average score, the score in the first test, and the score in the last test for each subject.

Input

A list of unsorted student records (student name, subject, test number, score). The last line contains the string `END` , do not process this line.

```
ADAM ENG 1 71
ADAM ENG 3 84
ADAM MAT 2 99
BETH SCI 1 79
ADAM ENG 2 90
BETH SCI 2 97
ADAM MAT 1 91
END
```

Output

For each student print: The student name, subject, average score (round to nearest integer), first score and final score.
The output should be sorted by the student name and subject.

```
ADAM ENG 82 71 84
ADAM MAT 95 91 99
BETH SCI 88 79 97
```

Cooking

Worth 3 point(s) - Runtime Limit: 1 seconds

Introduction

It’s time to bake some cookies for the holidays. You have a cookie recipe that bakes 22 cookies. It requires 3 cups of flour, 2 eggs, 2 cups of milk, and 1 cup of sugar to make the batch. Given this recipe and the amount of each ingredient that you possess, compute the maximum number of full-sized cookies that can be baked.

Input

The first line of input will contain an integer T denoting the number of test cases to follow. For each test case, there will be four space-separated integers, F, E, M, and S, giving the amount of each ingredient that you have. F gives the number of cups of flour, E the number of eggs, M the number of cups of milk, and S the number of cups of sugar.

```
2
7 4 3 6
2 2 3 2
```

Output

For each test case, output a line containing the maximum number of cookies that can be baked. Note that you will not be baking in any sort of batches but that you will be adjusting the recipe to maximize your cookie output.

```
33
14
```

The Very Last Minute

Worth 3 point(s) - Runtime Limit: 1 seconds

Introduction

Whoops! Someone designed this problem at the very last minute. Cutting it close! But how close?

Given a deadline and a starting time, write a program that finds the difference in hours and minutes.

Input

Input starts with a line containing N, the number of test cases. Each test case consists of two lines, one for a deadline and one for a starting time. The deadline will always fall after the starting time. Each line consists of a time-stamp (in military time), and day of the month separated by a space.

```
3
9:00 25
7:20 25
16:00 13
16:00 12
2:00 14
22:00 2
```

Output

Print out the time difference between deadline and starting time in hours and minutes.

```
1 hours 40 minutes
24 hours 0 minutes
268 hours 0 minutes
```

High Arctic Camel Case

Worth 3 point(s) - Runtime Limit: 1 seconds

Introduction

Normal camel case looks like this:

```
thisIsCamelCase
```

For every word in a sentence all spaces are removed and each word after the first starts with a capital letter.

But high arctic camels are different:

```
THISiShIGHaRCTICcAMELcASE
```

It is similar to normal camel case except that now the first letter is the only letter that is lowercased.

Given a string where words are separated by spaces, convert it to high arctic camel case.

Input

N lines containing one string needed to be converted to high arctic camel case. Input ends with one line containing `END`, do not process this line

```
Hello World
gOoD sAmPlE iNpUt rIgHt
END
```

Output

The high arctic camel case version of the given input strings.

```
HELLOwORLD
GOODsAMPLEiNPUTrIGHT
```

Removing Snow

Worth 3 point(s) - Runtime Limit: 3 seconds

Introduction

Congratulations! You own the Snow Remover 9000!

It can remove any size rectangle of snow in one fell swoop. Unfortunately, it can only be used once, so we'll have to maximize the efficiency of our snow removal. Given the sizes of the rows of plots on our land, find the size of the largest rectangle we can remove with the snow removal machine.

Input consists of a number N indicating the number of plots. Each plot starts with a number M indicating how many rows are in that plot, followed by M rows of numbers which correspond to the number of spaces on that row. As an example, the plot:

```
3
5
3
```

means a plot that looks like this:

```
***
*****
***
```

Find the largest rectangle of snow on the plot that you can clear with one run of your machine. For the sample plot above, that's 9.

Input

Input starts with N representing the amount of test cases. Each test case starts with one line containing M (where $0 \leq M \leq 1000$) and M lines afterwards.

NOTE: There will be no empty lines in the actual test cases. The ones below are used for the sake of brevity.

```
2

3
3
5
3

3
1
2
3
```

Output

Each line of output contains the largest area of the rectangle for each test case.

```
9
4
```

"No Two Snowflakes Are Alike" Is A Myth Actually

Worth 3 point(s) - Runtime Limit: 1 seconds

Introduction

Remember the thing where people think that no two snowflakes are alike? Did you know that's not true? ...No? Well maybe it is, but I'm pretty sure there was someone who found two snowflakes that were exactly the same

Input consists of a string of digits that corresponds to a snowflake pattern. Output the repeated snowflake pattern.

Input

Input consists of N test cases, where $0 \leq N \leq 2000$. Each test case contains one line with a number T, where $0 \leq T \leq 10^{30}$. The last line of input contains a `0`, do not process this line

```
1234567890
390543112
103827421287553411369671120
43509524942590534
1234567890
88888888888888888888
390543112
0
```

Output

Output the repeated snowflake pattern for each test case.

```
1234567890
390543112
```

Typing With a Flip Phone

Worth 3 point(s) - Runtime Limit: 1 seconds

Introduction

Nix has a flip phone and he uses that to text. If you don't know, a flip phone consolidates all of the alphabet in 9 digits requiring you to press a pattern of digits to get one character. For example the number 2 has the letters ABC above it and if you want to type a C, you have to press the 2 button 3 times to cycle A->B->C->2, and like the number 7 also has the letters PQRS above it and if you want to type an S, you have to press the 7 button 4 times to cycle between P->Q->R->S->7

Unfortunately, Nix's flip phone is messed up: the letters mapping to each of the buttons are weird. 0 still maps to space, but all the other numbers are different.

Input

Take an input consisting of 9 lines of keymappings and then one line containing a string Nix needs to type. The string that Nix needs to be typed will not contain any characters not in the 9 lines of keymappings, other than space, which is always mapped to 0, or numerals, which come after the letters in every cycle. Input terminates with a `#`.

```
ABC
DEF
GHI
JKL
MNO
PQRS
TUV
WXYZ
.,?!
HELLO WORLD!
QWE
RTY
UIOP
ASD
FGH
JKL
ZXC
VBN
M,.
L33T SP34K
#
```

Output

Determine how many button presses are required to type Nix's input string.

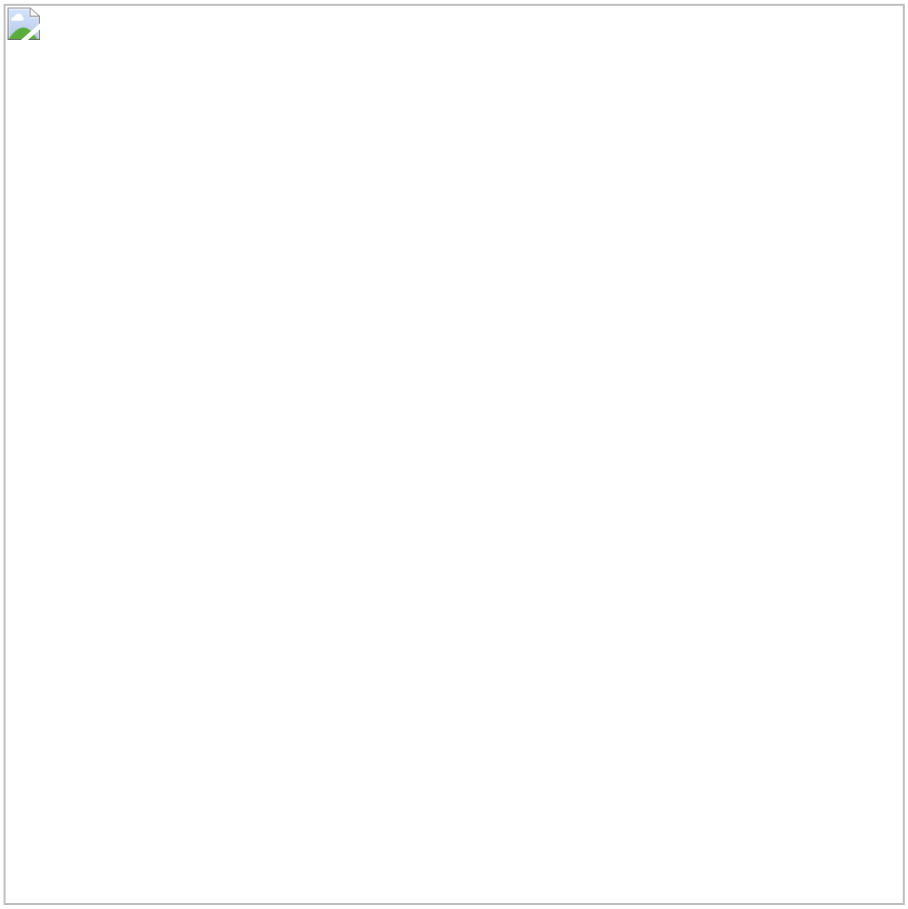
```
29
33
```

Automatic Ball Machine

Worth 2 point(s) - Runtime Limit: 3 seconds

Introduction

A tennis ball machine sits on the center mark of a tennis court, as shown below. It shoots balls randomly on the other side of the court. The position of the ball machine is (0, 0):



Write a program to find the distance from the ball machine to a given point on the court.

Input

Read the input one line at a time. Each line contains an (x, y) coordinate. The last line of input contains a `0 0`, do not process this line

```
3 4
6 8
8 8
2 3
0 0
```

Output

Find the distance from the ball machine to a given cordinate (x, y) on the court, rounded to two decimal places.

```
5.00
10.00
11.31
3.61
```

Frosty

Worth 2 point(s) - Runtime Limit: 1 seconds

Introduction

Ever heard of Frosty the Snowman? He’s like a winter Frankenstein’s monster, product of some children’s handiwork and a magical top hat. Let’s depict this legend with a bit of ascii art.

For this problem there will be no input.
Print out the ascii art of a snowman exactly as below.

Sample Output

```
   _[_]_
    (")
'--( : )--'
  (  :  )
""'-...-'""
```

Pie

Worth 2 point(s) - Runtime Limit: 1 seconds

Introduction

Alan Turing, Ada Lovelace, and Edsger Dijkstra are all sitting around a table, discussing cutting-edge computer science algorithms. Suddenly everyone gets hungry, and they decide to go out and buy a delicious peppermint pie. After returning from the grocery store with their pie, they open the box to find their pie cut into X slices. They realize that once they divide up the slices of pie between the three of them, there might be slices left over. Turing, Lovelace, and Dijkstra are still busy talking, but you can help them by writing a program to determine how many left over slices they will have after dividing the pie amongst themselves.

Sample Input

There will be multiple lines of input, each containing an integer X, the number of slices the pie has been cut into.

```
6
13
1
23
```

Sample Output

For each input line, output the number of slices which will have to be wrapped up and put in the fridge for leftovers.

```
0
1
1
2
```

The Biggest

Worth 2 point(s) - Runtime Limit: 1 seconds

Introduction

Everything’s bigger in Texas. But what’s the biggest? Let’s write a program!

Input

The first line of input will contain an integer T which tells how many numbers will follow and be compared. The next T lines will each contain an integer to be compared to the others. Keep in mind that they could be negative!

```
5
0
3
1
-50
2
```

Output

For output, print out the biggest number given!

```
3
```

Problem Credits

The following students have created problems and/or contributed to the competition problems:

Emily Peterson - CS Club President at Centennial High School
Matthew Zhang - CS Competition Manager at Reedy High School
Saif Ahmad - CS Event Manager at Reedy High School