**Problem 2.1** Print Sheep!

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

In the Land of Magic Starfalls, there aren't enough sheep to please the hungry mountain ogre Campbell. We desperately need more sheep on our ranch! Please help us get more sheep to feed the ogre.

**Output Information**

The output is to be formatted exactly like the output below.

**Sample Output**

\_\_

-('')-

%%%%%%

~%%%%%%

// ||

**Problem 2.2** Bell Call

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

You need to feed the sheep! You need to ring the bell 7 times, no more, no less. 7 will be the number of rings, and the number of rings will be 7. 6 is too few and 8 is too many. Ring the bell 7 times and 7 times only or else Nelson the Gold-Digger will steal your monies. When the bell has been rung, the sheep will eat, and all will be good in the Land of Magic Starfalls.

**Output Information**

Your program should output the ASCII character 0x07 seven times on one line. So, in total 7 characters. You may not be able to see the characters printed, but you should hear the computer beep (if it has a built-in speaker).

**Problem 2.3** Animal Inventory

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

Once upon a time, in the Land of Magic Starfalls, Lord Mary the Count wanted to learn her numbers. Thus, Sam the Rich Texan Farmer agreed to hire out his farm to her so that she may count all the animals and gain math success.

Write a program to check her checking or else Lord Mary the Count will impale you.

**Input Information**

The first line of the input is an integer *n* that represents the number of data collections that follow where each data collection is on a single line. Each data collection contains number of each animal, followed by the type of animal.

**Name of Data File**

pr23.dat

**Output Information**

Your program should output the total number of animals that were counted.

The output is to be formatted exactly like that for the sample output below.

**Assumptions**

The value of *n* will be between 1 and 100, inclusive.

**Sample Input**

4

5 cows

10 horses

15 camels

7 pigs

**Sample Output**

37 animals

**Problem 2.4** Parrot

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

The parrot of the Land of Magic Starfalls named Eisaundra has just flown away, and Franeus misses it dearly. Every day, Franeus would talk to the parrot, and it would repeat the first and last word of his sentence. Upon hearing of Franeus' sadness, you decide to write a program that would simulate his conversations with the parrot. Given that each individual sentence is on a separate line, print out the first and last word of each sentence.

**Input Information**

The first line of the input is an integer *n* that represents the number of data collections that follow where each data collection is on a single line. Each data collection contains a single string.

**Name of Data File**

pr24.dat

**Output Information**

Your program should produce *n* lines of output (one for each data collection).

The output is to be formatted exactly like that for the sample output given below.

**Assumptions**

The value of *n* will be between 1 and 100, inclusive.

**Sample Input**

3

I'm training myself to be more compassionate.

I rolled my eyes reflectively ceilingward and repeated the question aloud.

I’d like to think of myself as a professional short-story writer.

**Sample Output**

I'm compassionate.

I aloud.

I'd writer.

**Problem 2.5** Thanks, Hanks

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

Hassan the Harlequin is very lazy and discovered a way to thank people using fewer characters than normal. This method consists of taking the first two letters of "Thanks", and appending all the letters of the noun starting at the first vowel. For example, "Thanks, Franklin" becomes "Thanklin". Help Hassan generate all the thanks he needs.

**Input Information**

The first line of the input is an integer *n* that represents the number of data collections that follow where each data collection is on a single line.

**Name of Data File**

pr25.dat

**Output Information**

Your program should produce *n* lines of output (one for each data collection).

The output is to be formatted exactly like that for the sample output given below.

**Assumptions**

The value of *n* will be between 1 and 100, inclusive.

Every name given will contain at least one vowel.

**Sample Input**

4

Thanks, Mary

Thanks, George

Thanks, Lee

Thanks, Andrew

**Sample Output**

Thary

Theorge

Thee

Thandrew

**Problem 2.6** Cumulative Sum

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

The cumulative sum of an integer is the sum of every integer between and including 1 to *x*. For example, the cumulative sum of 5 is 15, because 1+2+3+4+5 = 15. Death Knight Franklin needs to calculate his runes. Therefore, your task is, given an integer *x*, calculate Franklin’s runes by taking the cumulative sum.

**Input Information**

The first line of the input is an integer *n* that represents the number of data collections that follow where each data collection is on a single line. Each data collection contains a single integer.

**Name of Data File**

pr26.dat

**Output Information**

Your program should produce *n* lines of output (one for each data collection). Each data collection should consist of the sum of the integers from 1 to *n*, inclusive.

The output is to be formatted exactly like that for the sample output given below.

**Assumptions**

The value of *n* will be between 1 and 100, inclusive.

Each integer from the data collection will be between 1 and 1000, inclusive.

**Sample Input**

4

8

64

512

1000

**Sample Output**

36

2080

131328

500500

**Problem 5.1** Animal Sort

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

The Ranch in the Land of Magic Starfalls has many animals, and they need to be ordered. The banker Andrew Ungalla needs a list of animals sorted by height in ascending order. If their height is the same, order them by weight.

The weight will be between 1 and 1500 pounds, inclusive.

The height will be between 0'1" and 6'0" inclusive.

**Input Information**

The first line of the input is an integer *n* that represents the number of data collections that follow where each data collection is on a single line. Each data collection contains the type of animal, the height in feet and inches, and the weight in pounds.

**Name of Data File**

pr51.dat

**Output Information**

Your program should produce *n* lines of output (one for each data collection). The output should list the animals sorted by height and weight.

The output is to be formatted exactly like that for the sample output given below.

**Assumptions**

The value of *n* will be between 1 and 100, inclusive.

**Sample Input**

4

Horse 5'4" 500

Cow 3'11" 500

Sheep 3'11" 400

Moose 5'4" 210

**Sample Output**

Sheep 3'11" 400

Cow 3'11" 500

Moose 5'4" 210

Horse 5'4" 500

**Problem 5.2** The Great Pizza in the Sky

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

In the Land of Magic Starfalls, a disaster has erupted! All of the stars in the sky have fallen! Rahul Ungalla, the owner of the local electronic shop, decides to take charge of this catastrophe and reassures the others that he remembers the position of every star in the sky. Upon saying this, he abruptly turns around and storms off. He has come to you for help, and he tells you of his plan to arrange the stars in the shape of a slice of pizza. Rahul Ungalla tells you that he will go back and count the numbers of rows that fell; in the mean time, write a program that will generate the star formation given the amount he counts.

**Input Information**

The input will contain a single integer representing how many rows the formation should be.

**Name of Data File**

pr52.dat

**Output Information**

Your program should produce *n* lines of output.

The output is to be formatted exactly like that for the sample output given below.

**Assumptions**

The value of *n* will be between 1 and 20, inclusive.

**Sample Input**

5

**Sample Output**

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

**Problem 5.3** Rabbit Multiplication

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

As Peter Rabbit would tell you, rabbits multiply very quickly. However, they don't multiply in the normal sense of the word. They multiply using their own method. For example, when a rabbit multiplies 234 x 19, the result is 2x1 + 2x9 + 3x1 + 3x9 + 4x1 + 4x9 = 90. Given two integers, multiply the integers using Peter Rabbit's technique.

**Input Information**

The input will be two integers, separated by a single space.

**Name of Data File**

pr53.dat

**Output Information**

Your program should output one integer.

**Assumptions**

Each integer given will be between 0 and 100000, inclusive.

**Sample Input**

234 19

**Sample Output**

90

**Problem 5.4** Message Decoding

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

While digging in the Land of Magic Starfalls for buried treasure, you come upon a buried cardboard box. Upon looking inside the box, you find a note. However, the content of the note was encoded in a way that you have never seen before. On the first line of the note was a single word. You assume this to be the key of the encoded message. You believe the remaining lines to be decoded in the following way.

Each character in the message is to be “added” to a character in the key. The index of the character in the key to be added increases with each index in the message that is an A-Z character, and if the index of the key goes over the length, it wraps around to the beginning. The way characters are added is based on the index in the alphabet (A=1, B=2, C=3, etc). A+A=B, A+B=C. If the total goes over 26, it wraps around to the beginning. A+Z=A, C+Y=B.

So for example, if the key is "ABC" and the message is "AA AAAB", the decoded message is

"BC DBCE".

Decode the message using the key, and return the output.

**Input Information**

The first line of the input is the key. The second line of the input is the message to be decoded.

**Name of Data File**

pr54.dat

**Output Information**

Your program should produce 1 line of output, which is the decoded message.

The output is to be formatted exactly like that for the sample output given below.

**Assumptions**

The key will only be the characters A-Z, all uppercase.

The decoded message will not contain lowercase characters.

Both the encoded and decoded message can contain numbers, spaces, and symbols; preserve these.

**Sample Input**

BISCUIT

KVLQ RV GR KOB YIGGE ZQFKOME HQ XZD ND VK SFBCDIBW KNC KDBSKE-RYPOI. UU LFA YJ CGRV.

**Sample Output**

MEET ME AT THE TRAIN STATION AT SIX PM ON NOVEMBER THE TWENTY-THIRD. DO NOT BE LATE.

**Problem 5.5** Engrish

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

Jonathan the Fat Duke in the Land of Magic Starfalls is writing a paragraph for his English class, and wants you to check over it. Upon receiving it, you were shocked by the extreme punctuation errors it contained.

You decide that there are two main things that need to be fixed: contractions need to be spelled out, and the first word of every sentence needs to be capitalized.

Specifically, the contractions *can't*, *don't*, and *won't* need to be spelled out; however, don't assume the apostrophe will be there!

Jonathan doesn't want to fail his class; his grade is up to you!

**Input Information**

The first line of the input is an integer *n* that represents the number of data collections that follow where each data collection is on a single line. Each data collection contains a single string.

**Name of Data File**

pr55.dat

**Output Information**

Your program should produce *n* lines of output (one for each data collection). Each data collection should be the corrected version of the sentence.

The output is to be formatted exactly like that for the sample output given below.

**Assumptions**

The value of *n* will be between 1 and 100, inclusive.

**Sample Input**

8

the norse gnome attacked and pillaged the pajama stand at the market.

the Umbrella in the stand that cant be red covered the wax candle in the corner of the room!

"god is the best" the gnome said and spiced up his stew.

he threw the umbrella into the sky and tried to impale his elephanto.

"don't stop believing" the gnome said to the peanut as he crashed his jelly into the car.

after the jelly hit the car the strawberry man yelled "SCOTCH TAPE!" at the loopy elf.

the molecular structure cell had lost bounds and threw an out of bounds exception at the man.

and so, the children learnt their lesson of not to take candy from strangers.

**Sample Output**

The norse gnome attacked and pillaged the pajama stand at the market.

The Umbrella in the stand that can not be red covered the wax candle in the corner of the room!

"God is the best" the gnome said and spiced up his stew.

He threw the umbrella into the sky and tried to impale his elephanto.

"Do not stop believing" the gnome said to the peanut as he crashed his jelly into the car.

After the jelly hit the car the strawberry man yelled "SCOTCH TAPE!" at the loopy elf.

The molecular structure cell had lost bounds and threw an out of bounds exception at the man.

And so, the children learnt their lesson of not to take candy from strangers.

**Problem 5.6** Easy as 123

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

Lydia the Plumber is trying to teach the animals at the Ranch of Magic Starfalls for some reason. Today's lesson is a function that no matter what number is given will always yield the same result: 123. The function is as follows. Given an integer *x*, count the number of even digits (this will be the first part of the result). Then, count the number of odd digits, and put this at the **end** of the number from before. Then, count the total number of digits, and put this at the **end** again. For example, if the number is 4005, there are 3 even digits, 1 odd digit, and 4 digits total, so the result is 314. If the number of even digits is 0, include that in your result (so 5 becomes 011). This function is repeated until the answer is 123. So, continuing the example, with 314 there is 1 even digit, 2 odd digits, and 3 digits total: 123.

Write a program that, given an integer *x*, prints the steps going from *x* to 123.

**Input Information**

The first line of the input is an integer *n* that represents the number of data collections that follow where each data collection is on a single line. Each data collection contains a single integer.

**Name of Data File**

pr56.dat

**Output Information**

Your program should produce *n* sets of output (one for each data collection). Each data collection should be separated by a single blank line.

The output is to be formatted exactly like that for the sample output given below.

**Assumptions**

The value of *n* will be between 1 and 100, inclusive.

The value of *x* will be between 0 and 1000000, inclusive.

**Sample Input**

3

5

42

39078

**Sample Output**

5

011

123

42

202

303

123

39078

235

123

**Problem 9.1**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

David the Pixie wants to get out of the Cave of Magic Starfalls. Assist him in any way possible. Death sheep pits will be located around the cave. Find the shortest path out of the cave without falling into a death sheep pit.

Entrance = O;

Exit = X;

Sheep =S;

Wall = E;

**Input Information**

The first line of the input is an integer *n* that represents the number of lines that follow. Each line represents a row of the maze.

**Name of Data File**

pr91.dat

**Output Information**

Your program should produce *n* lines of output.

The output is to be formatted exactly like that for the sample output given below.

**Assumptions**

The value of *n* will be between 5 and 20, inclusive.

**Sample Input**

EEEEEEEE

O ES E

E E E E

E E E

E E EE E

E SE E

EEEEEXEE

**Sample Output**

EEEEEEEE

O\*\*\*ES E

E E\*\*E E

E E\*\*\*E

E E EE\*E

E SE \*\*E

EEEEEXEE

**Problem 9.2** Animal Alphabet

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

The animals in the Zoo of Magic Starfalls need education too. In fact, the Peter Liang School of Computer Science has a campus on site specifically for the edumacation of the animals. Unfortunately, a mistake was made, and the animals did not learn the digit 4. Therefore, whenever the animals count, they skip over any number that has a 4 in it. For example, when a hedgehog says he has 6 apples. He really only has 5 apples. When a platypus says he swam 15 miles yesterday, he really swam only 13 miles. Given numbers written in this fashion, print what the values of the numbers really are.

**Input Information**

The first line of the input is an integer *n* that represents the number of data collections that follow where each data collection is on a single line. Each data collection contains a single integer.

**Name of Data File**

pr92.dat

**Output Information**

Your program should produce *n* lines of output (one for each data collection).

The output is to be formatted exactly like that for the sample output given below.

**Assumptions**

The value of *n* will be between 1 and 100, inclusive.

Each integer in the data set will be between 0 and 100000, inclusive.

**Sample Input**

4

3

6

15

999

**Sample Output**

3

5

13

728

**Problem 9.3** Interpreter

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

Hassan the Harlequin has been struggling with the concepts of Java for a while now, and out of this frustration decided to create his own programming language.

The language contains only eight instructions. For this language, each instruction is one character long, and every character that is not an instruction is ignored. The language has a 10,000 cell memory stored in an array, each cell is one byte, and the data is accessed via a pointer. The following characters are the instructions:

**>** Moves the pointer position right one cell

**<** Moves the pointer position left one cell

**+** Increments the memory cell at the pointer position

**-** Decrements the memory cell at the pointer position

. Prints the ASCII character from the cell at the pointer

, Reads in a character and sets the ASCII value to the cell at the pointer

**(** If the cell at the pointer is 0, jump to the matching )

**)** If the cell at the pointer is NOT 0, go back to the matching (

Write an interpreter for Hassan's language that reads in the instruction string and prints the output.

**Input Information**

The first line of the input is a string which is the instructions to be executed, followed by the input data for that string on the next line.

**Name of Data File**

pr93.dat

**Output Information**

The output is to be whatever the interpreted string outputs.

**Assumptions**

The initial value of the pointer is 0.

The pointer will not go below 0 or above 9,999.

The cell value will not go below 0 or above 255.

There will always be enough input data.

Each '(' will have a matching ')'

**Sample Input**

>,-.,+.(-)+++++>,<(>.+<-)

ba0

**Sample Output**

ab01234

**Problem 9.4** Tree Ordering

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

Oh noes! Peter Rabbit's natural habitat is being demolished by civilization. He would like to rebuild the trees that have being cut down. Peter Rabbit's trees must be built in post-order fashion. However, Peter Rabbit only knows the pre-order and in-order traversals of the trees.

Pre-order traversal of a binary tree prints the path in the following order: Parent, Left Child, Right Child.

In-order traversal of a binary tree prints in the following order: Left Child, Parent, Right Child.

Post-order traversal prints in the following order: Left Child, Right Child, Parent.

Given the in-order and pre-order traversals, only one distinct tree is possible. Print the post-order traversal of this tree. For example, if pre-order is 5 2 1 4 8 9 and in-order is

1 2 4 5 8 9, then the tree would look like:

5

/ \

2 8

/ \ \

1 4 9

The post-order traversal of this tree would be: 1 4 2 9 8 5.

If pre-order is 8 4 2 1 9 5 and in-order is 1 2 4 5 8 9, the tree would look like:

8

/ \

4 9

/ \

2 5

/

1

Post order would be 1 2 5 4 9 8.

**Input Information**

The first line of the input is an integer *n* that represents the number of data collections that follow where each data collection is on two lines. Each data collection contains the pre-order, followed by the in-order.

**Problem 9.4** Tree Ordering (cont.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Name of Data File**

pr94.dat

**Output Information**

Your program should produce *n* lines of output (one for each data collection).

The output is to be formatted exactly like that for the sample output given below.

**Assumptions**

The values of the nodes in the tree will be distinct. The amount of nodes in the tree will be less than 30.

**Sample Input**

2

5 2 1 4 8 9

1 2 4 5 8 9

8 4 2 1 9 5

1 2 4 5 8 9

**Sample Output**

1 4 2 9 8 5

1 2 5 4 9 8

**Problem 9.5** Happy Cats

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

Peter Rabbit is farming cats, or rather he is pharming cats. He doesn't want sad cats. He prefers for his cats to be happy. Two cats are happy if and only if they meet the following condition. You are given the endpoints of a fence. You are also given the coordinate points of two cats. Each pairs of cats are happy only if they cannot see each other. This happens when the fence blocks the line of sight between the cats. In other words, let V be the line formed by using the location of the cats as endpoints. If V intersects with the fence, then the cats cannot see each other. If that isn't clear enough, then I'll make it even simpler: We want to figure out whether two lines intersect or not. You will not be given any input where an endpoint of one line lies on another line.

If two cats are happy, print out "HAPPY". If they are not happy, print out "NOT HAPPY".

**Input Information**

The first line of the input is an integer *n* that represents the number of data collections that follow where each data collection is on two lines. The first line of the data set is the fence endpoints in X Y X Y format. The second line of the data set is the coordinates of each cat in X Y X Y format.

**Name of Data File**

pr95.dat

**Output Information**

Your program should produce *n* lines of output (one for each data collection).

The output is to be formatted exactly like that for the sample output given below.

**Assumptions**

The value of *n* will be between 1 and 100, inclusive.

**Sample Input**

2

-4 -7 -4 8

-7 -5 -2 8

-1 -12 4 5

-2 -8 6 5

**Sample Output**

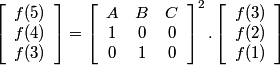
HAPPY

NOT HAPPY

**Problem 9.6** Peter Rabbit Problem

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General Statement**

Peter Rabbit would like to solve linear recurrences. The linear recurrence he is trying to solve is in the form **f(x) = A \* f(x-1) + B \* f(x-2) + C \* f(x-3)**.  
However, being Peter, he does not want to solve it that way as it is too mundane. Instead, he would like to use the method of matrix exponentiation. He notices that with the following relations:  
  
http://c2008.team-duck.com/files/pr96_01.gif  
  
He can setup a nice matrix equation.  
  
http://c2008.team-duck.com/files/pr96_02.gif  
  
          K              =             M              x             A  
  
Now, given constants **f(1)**, **f(2)**, **f(3)**, **A**, **B**, **C**, we can solve for **f(x) | x > 3** in a neat manner. Since matrix multiplication is associative, we can simply raise the transformation matrix **M** to the **j-th** power, where **j = x-3**, then multiply the result to vector **A**. The resulting vector **K** should then contain the elements **f(x), f(x-1), f(x-2)**.  
  
For example, to find **f(5)**, we simply do:  
  
  
  
Now, Peter doesn’t want you to cheat, and just do it the normal way, so not only do you have to output **f(x)**, but you will also have to output the 3 x 3 matrix **M^j**.

**Input Information**

The first line contains a number *n*, the number of test cases that follow. Each test case consists of 2 lines. The first line contains a single integer *x*, the second line contains 6 integers, A, B, C, f(1), f(2), f(3) respectively.

**Name of Data File**

pr96.dat

**Problem 9.6** Peter Rabbit Problem (cont.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Output Information**

You should output **f(x)** followed by the 3 x 3 matrix **M^j** with a space between each element. There should be NO spaces between each set out of outputs.

**Assumptions**

**3 < x < 10** and **1 <= A <= B <= C <= f(1) <= f(2) <= f(3) <= 6**. You should not worry about integer overflow with the matrix multiplication.

**Sample Input**

2

4

1 1 1 1 1 1

5

1 2 3 3 2 1

**Sample Output**

3

1 1 1

0 1 0

0 0 1

22

3 5 3

1 2 3

1 0 0