1. Cab Numbers

Program Name: CabNumbers.java   Input File: cabnumbers.dat

This is a story related by the mathematician Hardy when he visited the Indian mathematician Ramanujan:

I remember once going to see him when he was ill at Putney. I had ridden in taxi cab number 1729 and remarked that the number seemed to me rather a dull one, and that I hoped it was not an unfavorable omen. "No," he replied, "it is a very interesting number; it is the smallest number expressible as the sum of two positive cubes in two different ways."

There are many such numbers less than 1,000,000. Write a program that will print a given number of the smallest positive integers that can be expressed as the sum of two positive cubes in two different ways. For example, here are the first three smallest numbers:

$1729 = 1^3 + 12^3 = 9^3 + 10^3$
$4104 = 2^3 + 16^3 = 9^3 + 15^3$
$13832 = 2^3 + 24^3 = 18^3 + 20^3$

Input

The input will be a single integer `n` that indicates the number of integers that you will print that meet the criteria above.

Output

You will print the first `n` smallest positive integers that can be expressed as the sum of two positive cubes in two different ways. Print one integer to a line in ascending order.

Example Input File

```
3
```

Example Output to Screen

```
1729
4104
13832
```

2. Call 555 - HELP

Program Name: Call.java   Input File: call.dat

Phone numbers are sometimes converted to mnemonics by advertisers and others to help remember numbers. The numbers are converted to letters based on some system that associates letters to numbers.

For this program, each digit (0-9) can be combined with any one of the letters associated with that digit. Write a program to generate all the possible mnemonics for a given number based on the letters associated with that digit. The number to be converted to a mnemonic will contain 12 or fewer digits. Each of the ten digits (0-9) will be associated with at least one letter and no letter will be associated with more than one digit.

Input

- The first line of the input file will contain a single integer `n` that indicates the number of data sets to follow.
- Each data set will consist of 11 lines.
  - The first line of each data set will contain a single integer with 12 or fewer digits to convert to a mnemonic.
  - Each of the next 10 lines will contain a single digit and a space followed by the lower case letters (a-z) to which that digit can be associated.

Output

For each data set output "`Mnemonics for <num>:`" where `<num>` is replaced by the number for that data set (the number to be converted to a mnemonic). Then display all possible mnemonics for the given number, one per line, in alphabetical order.

Example Input File

```
2
55
0 xyz
1 ab
2 cd
3 efg
4 hi
5 jk
6 lmn
7 opq
8 rst
9 uvw
213
5 b
3 klm
1 i
0 zd
9 yun
2 cua
8 e
7 f
6 trs
4 f
```

Example Output To Screen

```
Mnemonics for 55:
jj
jk
kj
kk
Mnemonics for 213:
aik
ail
aim
cik
cil
cim
uik
uil
uim
```

3. The Count

Program Name: Count.java   Input File: count.dat

The Count from Sesame Street got his name because he counts everything that he sees. You are to write a program for The Count that will count the number of lines in an input file and print that number to the screen.

Input
The input file contains an unknown number of lines.

Output
Print the count of the number of lines in the input file written as a single integer.

Example Input File
```
Mary had a little lamb
Jack be nimble, Jack be quick
Hickory, Dickery, Dock
```

Example Output To Screen
```
3
```

4. Mad-Lib

Program Name: MadLib.java Input Files: madlib.dat

Just for fun, Michael wants you to write a Mad-Lib program. He has designed it so he will read a list of words from the beginning of a file and then insert those words into the story that follows.

Input

The first line will contain a single integer `n` that indicates the number of mad-libs to follow. For each mad-lib, the first line will contain a single integer `m`. Each of the next `m` lines will contain a single item consisting of one or more words. Each of the next `m+1` lines will contain the text of the story.

The items will be inserted between the end of the text of one line and before the beginning of the text of the next line, in the order they appear to form one paragraph.

Output

For each mad-lib, print the complete mad-lib story with the items properly inserted. The story will be written in one paragraph with no more that 40 characters on a line and as many whole words (including contiguous punctuation) on a line as possible. There must be a space on either side of the text that is inserted into the mad-lib to separate it from the text in the story except that no space will appear if the item is inserted before a period or a comma. Place some whitespace after each paragraph of output.

Example Input

```
1
6
car race
sweater
giraffe
jumped
ear
cake
Jerry came to me in the middle of the
and wanted me to study for a
in Computer Science. I told him that I had a
and could not. Since then, I
and changed my
. Studying is the most important
that I do.
```

Example Output To Screen

```
Jerry came to me in the middle of the
car race and wanted me to study for a
sweater in Computer Science. I told him
that I had a giraffe and could not.
Since then, I jumped and changed my ear.
Studying is the most important cake that
I do.
```

5. Magic Trick

Program Name: Trick.java   Input File: trick.dat

At the Math Club's annual banquet, the after-dinner speaker asked one of the students to give him an integer between 34 and 50. The student said 43. The speaker then created a 4 x 4 square in which each column, each row, and each diagonal summed to the magic number 43. This kind of square is called a magic square.

```
 8 11 23  1
22  2  7 12
 3 25  9  6
10  5  4 24
```

After some investigation and discussion, one member of the club discovered the speaker's trick. The numbers 1 through 12 are preset in certain places as shown in the square to the right. The remaining four numbers are consecutive integers and will always begin with 21 less than the magic number. In this case, 43 – 21 = 22 so the numbers 22, 23, 24, and 25 are placed in the remaining four positions.

You will be given a magic number and a 4 x 4 square with the numbers 1 through 12 already preset. You are to write a program to determine the position of the remaining four numbers so that the square meets all the above criteria for a magic square.

Input

The first line will contain a single integer `n` that indicates the number of data sets to follow. The first line of each data set will contain the magic number and will be followed by a set of 4 lines that contain a 4 x 4 square. The square will contain the numbers 1 through 12 and asterisks (`*`) to indicate where the remaining four integers will be placed.

Output

For each magic number and 4 x 4 square, create a magic square that meets all of the criteria above by replacing the asterisks (`*`) with the correct numbers. Print whitespace between each magic square.

Example Input File

```
2
43
8 11 * 1
* 2 7 12
3 * 9 6
10 5 4 *
42
* 1 12 7
11 8 * 2
5 10 3 *
4 * 6 9
```

Example Output to Screen

```
8 11 23 1
22 2 7 12
3 25 9 6
10 5 4 24

22 1 12 7
11 8 21 2
5 10 3 24
4 23 6 9
```

6. Stars

Program Name: Stars.java   Input File: none

Jeremy wants to print a design of asterisks (*) to use as a pattern for an art project. You are to print his design as shown below.

Input

There is no input for this problem.

Output

You are to output a triangle of asterisks (*) with 20 rows as shown below.

Output to Screen

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