

## Problem A

### Let's get started!!!!

Source Files : A.cpp, A.pas, A.java  
Input File : A.in      Output File : A.out

This problem is just a warm up problem, reading from the input file and writing in the output file, just this.

#### Input

The input file starts with the integer n on a line by itself this is the number of strings to follow. The following n lines each contain one string of at most 50 letters .

#### Output

For each string in the input, first output the number of the string, as shown in the sample output. Then print the string that is read from the input. Print a blank line after each string.

#### Sample Input:

```
2
HAL
SWERC
```

#### Sample Output:

```
String #1
HAL

String #2
SWERC
```

### Problem B Reorder it

Source Files : *B.cpp, B.pas, B.java*  
Input File : *B.in*      Output File : *B.out*

You will be given three integers A, B and C. The numbers will not be given in that exact order, but we do know that A is less than B and B is less than C.

#### Input

The first line contains three positive integers A, B and C, not necessarily in that order. All three numbers will be less than or equal to 100.

The second line contains three uppercase letters 'A', 'B' and 'C' (with no spaces between them) representing the desired order.

#### Output

Output the A, B and C in the desired order on a single line, separating by single spaces.

#### Sample Input:

```
6 4 2  
CAB
```

#### Sample Output:

```
6 2 4
```

## Problem C Triangular numbers

Source Files : C.cpp, C.pas, C.java  
Input File : C.in      Output File : C.out

The  $n^{th}$  Triangular number,  $T(n) = 1 + \dots + n$ , is the sum of the first  $n$  integers. It is the number of points in a triangular array with  $n$  points on side. For example  $T(4)$ :

```
  X
 X X
X X X
X X X X
```

Write a program to compute the weighted sum of triangular numbers:

$$W(n) = \sum_{k=1}^n k * T(k + 1)$$

### Input

The first line of input contains a single integer  $N$ , ( $1 \leq N \leq 1000$ ) which is the number of datasets that follow. Each dataset consists of a single line of input containing a single integer  $M$ , ( $1 \leq M \leq 300$ ), which is the number of points on a side of the triangle.

### Output

For each dataset output on a single line the dataset number, (1 through  $N$ ), a blank, the value of  $M$  for the dataset, a blank and the weighted sum,  $W(M)$  of triangular numbers for  $n$ .

### Sample Input:

```
4
3
4
5
10
```

### Sample Output:

```
1 3 45
2 4 105
```

3 5 210

4 10 2145

### Problem D Scramble Sort

Source Files : *D.cpp, D.pas, D.java*  
Input File : *D.in*      Output File : *D.out*

In this problem you will be given a series of lists containing both words and numbers. The goal is to sort these lists in such a way that all words are in alphabetical order and all numbers are in numerical order. Furthermore, if the  $n^{\text{th}}$  element in the list is a number it must remain a number, and if it is a word it must remain a word.

#### Input

The input will contain multiple lists, one per line. Each element of the list will be separated by a comma followed a space, and the list will be terminated by a period. The input will be terminated by a line containing only a single period.

#### Output

For each list in the input, output the scramble sorted list, separating each element of the list with a comma followed by a space, and ending the list with a period.

#### Sample Input:

```
0.  
banana, strawberry, OrAnGe.  
Banana, StRaWbErRy, orange.  
10, 8, 6, 4, 2, 0.  
x, 30, -20, z, 1000, 1, Y.  
50, 7, kitten, puppy, 2, orangutan, 52, -100, bird, worm, 7, beetle.  
.
```

#### Sample Output:

```
0.  
banana, OrAnGe, strawberry.  
Banana, orange, StRaWbErRy.  
0, 2, 4, 6, 8, 10.  
x, -20, 1, Y, 30, 1000, z.  
-100, 2, beetle, bird, 7, kitten, 7, 50, orangutan, puppy, 52, worm.
```

### Problem E Doing Windows

Source Files : *E.cpp, E.pas, E.java*  
Input File : *E.in*      Output File : *E.out*

The screen of monitors on computer systems are rectangles. The aspect ratio of a screen is its width divided by its height. This term can also be applied to rectangular windows that may appear on the monitor's screen, where it is defined as the width of the window divided by its height. For this problem we assume the dimensions of a monitor's screen and its windows are measured in integral numbers of pixels, the individual dots (arranged in a rectangular grid) that comprise and image.

Suppose your windowing software only allows windows to be resized in such a way that their aspect ratios are unmodified. For example, a window with a width of 150 pixels and a height of 100 pixels (and an aspect ratio of 150/100, or 1.5) can be resized so its width is 225 pixels and its height is 150 pixels (the aspect ratio remains unchanged, at 225/150, or 1.5), but a width of 224 and a height of 150 is not allowed, since that would change the aspect ratio. Each window can be moved to an arbitrary location on the screen, but the entire window must remain visible on the screen.

Given the size of a screen and the initial sizes of four different windows (as integer values for width and height), is it possible to resize (and relocate) the four windows so they completely cover the screen without overlapping each other? That's the question you are to answer in this problem.

For example, consider a square screen with four square windows. The aspect ratio for each of these is exactly 1. We are permitted to resize each of the four windows so they would completely fill the screen without overlapping. This case is illustrated by the first data set in the Sample Input, below.

#### Input

Input will consist of one or more data sets followed by a pair of zeroes. Each data set will contain five pairs of integers. The first pair ( $W_s, H_s$ ) specifies the width and height of the screen. The four remaining pairs ( $W_i, H_i$ , for  $i = 1$  to 4) specify the initial sizes of the windows.

#### Output

Output should have one line for each input data set. The line should contain the input data set number (starting with 1) followed by the word Yes if the screen can be completely covered by the (possibly resized and relocated) windows with no overlap, or No if it cannot be so covered.

### Sample Input:

```
400 400 10 10 35 35 15 15 100 100
200 300 10 10 20 20 30 45 40 60
200 250 10 10 20 20 30 45 40 60
0 0
```

### Sample Output:

```
Set 1: Yes
Set 2: No
Set 3: Yes
```

### Problem F Extrapolation Using a Difference Table

Source Files : *F.cpp, F.pas, F.java*  
Input File : *F.in*      Output File : *F.out*

A very old technique for extrapolating a sequence of values is based on the use of a difference table. The difference table used in the extrapolation of a sequence of 4 values, say 3, 6, 10, and 15, might be displayed as follows:

3			
6	3		
10	4	1	
15	5	1	0

The original sequence of values appears in the first column of the table. Each entry in the second column of the table is formed by computing the difference between the adjacent entries in the first column. These values (in the second column) are called first differences. Each entry in the third column is similarly the difference between the adjacent entries in the second column; the third column entries are naturally called second differences. Computation of the last column in this example should now be obvious (but beware that this value is not always zero). Note that the last column will always contain only a single value. If we begin with a sequence of  $n$  values, the completed difference table will have  $n$  columns, with the single value in column  $n$  representing the single  $n-1$ st difference.

To extrapolate using a difference table we assume the  $n - 1^{st}$  differences are constant (since we have no additional information to refute that assumption). Given that assumption, we can compute the next entry in the  $n - 2^{nd}$  difference column, the  $n - 3^{rd}$  difference column, and so forth until we compute the next entry in the first column which, of course, is the next value in the sequence. The table below shows the four additional entries (in boxes) added to the table to compute the next entry in the example sequence, which in this case is 21. We could obviously continue this extrapolation process as far as desired by adding additional entries to the columns using the assumption that the  $n - 1^{st}$  differences are constant.

3				
6	3			
10	4	1		
15	5	1	0	
21	6	1	0	

### Input

The input for this problem will be a set of extrapolation requests. For each request the input will contain first an integer  $n$ , which specifies the number of values in the sequence to be extended. When  $n$  is zero your program should terminate. If  $n$  is non-zero (it will never be larger than 10), it will be followed by  $n$  integers representing the given elements in the sequence. The last item in the input for each extrapolation request is  $k$ , the number of extrapolation operations to perform; it will always be at least 1. In effect, you are to add  $k$  entries to each column of the difference table, finally reporting the last value of the sequence computed by such extrapolation. More precisely, assume the first  $n$  entries (the given values) in the sequence are numbered 1 through  $n$ .

### Output

Your program is to determine the  $n + k^{th}$  value in the sequence by extrapolation of the original sequence  $k$  times. (Hint: no upper limit is given for  $k$ , and you might not be able to acquire enough memory to construct a complete difference table.)

### Sample Input:

```
4 3 6 10 15 1
4 3 6 10 15 2
3 2 4 6 20
6 3 9 12 5 18 -4 10
0
```

### Sample Output:

```
Term 5 of the sequence is 21
Term 6 of the sequence is 28
Term 23 of the sequence is 46
Term 16 of the sequence is 8412
```



## Problem G Barcode

Source Files : G.cpp, G.pas, G.java  
Input File : G.in      Output File : G.out

Barcode consists of black and white vertical bars transformed to ones and zeroes by an optical reader. White and black bars alternate and can be thick or thin. A thin bar represents 0 and a thick bar represents 1, regardless of color. Thus, each barcode represents a sequence of digits.

Each bar in a barcode of a product appears as five “squares” high column. The width of a thin bar is one and the width of a thick bar is two “squares”.

The reader used in this problem fell on the floor and since then it was unable to properly recognize the color of some “squares” of a barcode.

Write a program that from a given scanning of a bar code with our faulty reader will determine the binary sequence if at all possible.

### Input

The first line of the input file contains an integer  $N$ ,  $1 \leq N \leq 100$ , representing the total width of the scanned barcode.

Each of the next five lines contains  $N$  characters, each of which can be either “X”, “.”(Dot), “?” (Question mark). ‘X’ represents successfully recognized black “square”, a dot represents successfully recognized white “square” and a question mark means that the reader couldn’t determine the color of the “square”.

### Output

The only line of the output should contain a sequence of binary digits without separators that represent the barcode or the word “UNDETERMINABLE” (without quotes) if a unique binary sequence cannot be determined.

### Sample Input:

```
4
.X??
.??
???.?
? X.?
.X?.
```

### Sample Output:

```
001
```

### Problem H John Locke

Source Files : *H.cpp, H.pas, H.java*  
Input File : *H.in*      Output File : *H.out*

John Locke is one of the survivors of flight 815. After the crash he found a hatch where he found that his destiny is to push a button every 108 minutes. But he also wants to go hunting boars, because he doesn't want to eat DHARMA INITIATIVE foods.

The problem is that he doesn't know when to come back from hunting, help him by writing a program that uses the computer and it's alarm to tell him how much time do he have for pushing that button.

#### Input

The first line is an integer number representing number of datasets, each dataset consists of 3 integers X, V, T. where X is the distance (*m*) between the place where John is looking for boars and the hatch, V is the maximum speed ( $\frac{Km}{H}$ ) that he can run and T is the time (*minutes*) left for pushing the button.

#### Output

One line for each dataset consisting of "John can save the world." or "Oops, NOooo.". "John can save the world." if he can return before T minutes to the hatch and push the button. "Oops, NOooo." if he can't return in T minutes and the world will be destroyed.

(Hint:  $T = \frac{x}{v}$ , u can use this formula to calculate time that John will be back at the hatch.)

#### Sample Input:

```
2
2000 10 2
100 10 12
```

#### Sample Output:

```
Oops, NOooo.
John can save the world.
```

### Problem I Prime Gap

Source Files : *I.cpp, I.pas, I.java*  
Input File : *I.in*      Output File : *I.out*

The sequence of  $n - 1$  consecutive composite numbers (positive integers that are not prime and not equal to 1) lying between two successive prime numbers  $p$  and  $p + n$  is called a prime gap of length  $n$ . For example, (24, 25, 26, 27, 28) between 23 and 29 is a prime gap of length 6.

Your mission is to write a program to calculate, for a given positive integer  $k$ , the length of the prime gap that contains  $k$ . For convenience, the length is considered 0 in case no prime gap contains  $k$ .

#### Input

The input is a sequence of lines each of which contains a single positive integer. Each positive integer is greater than 1 and less than or equal to the  $100000^{th}$  prime number, which is 1299709.

The end of the input is indicated by a line containing a single zero.

#### Output

The output should be composed of lines each of which contains a single non-negative integer. It is the length of the prime gap that contains the corresponding positive integer in the input if it is a composite number or 0 otherwise. No other characters should occur in the output.

#### Sample Input:

```
10
11
27
2
492170
0
```

#### Sample Output:

```
4
0
6
0
114
```