UNIVERSIDAD DISTRITAL
FRANCISCO JOSE DE CALDAS

MARATON SEMANA TECNOLOGICA

CATEGORIA AVANZADOS

UNIVERSIDAD DISTRITAL

11 de Noviembre de 2009

PROBLEMAS

Elaborado por: Hector Florez
Basado de www.acis.org.co

Problem 1.

Rafting

Source file name: rafting.c, rafting.cpp or rafting.java
Input: rafting.in
Output: standard output

A whitewater rafting company is trying to fit its clients into as few rafts as possible. Each raft can take one or two people and has a weight limit. You will be given the weight limit for rafts and the list of clients' weights. Compute the minimum number of rafts needed to accommodate all clients.

Input

The first line of input contains the number of test cases. The test cases follow. Each test case consists of two lines. The first line contains two integers n and w, 1 <= n; w <= 1000. n is the number of clients, w is the weight limit of each raft. The second line contains n integers between 1 and 1000, the weights of the clients.

Output

For each test case output a single line to the standard output. It should contain the minimum number of rafts or the word IMPOSSIBLE if no assignment is possible.

Sample input	Sample Output
2 3 100 95 13 25 3 100 1000 1000 1000	2 IMPOSSIBLE

Problem 2.

Pyramid

Source file: pyramid.c, pyramid.cpp o pyramid.java
Input: pyramid.in
Output: standard output

Input

The input contains several test cases. Each line describes a case. Each case contain a number 1<=n<=20 which indicate the high of the pyramid. The end of the input is indicated by a line that contains a zero.

Output

For each case, you must print n rows. Each row has the number i that is equivalent to the number of the row repeated i times building a pyramid.

Sample input	Sample Output
1 2 4 0	1 1 22 1 22 333 4444

Problem 3.

Count on Cantor

Source file: cantor.c, cantor.cpp o cantor.java
Input: cantor.in
Output: standard output

One of the famous proofs of modern mathematics is Georg Cantor's demonstration that the set of rational numbers is enumerable. The proof works by using an explicit enumeration of rational numbers as shown in the diagram below.

1/1 1/2 1/3 1/4 1/5 . . .
2/1 2/2 2/3 2/4 . . .
3/1 3/2 3/3 . . .
4/1 4/2 . . .
5/1 . . .

In the above diagram, the first term is 1/1, the second term is 1/2, the third term is 2/1, the fourth term is 3/1, and the fifth term is 2/2, and so on. You are to write a program that will read a list of numbers and will print for each number the corresponding term in Cantor's enumeration as given below.

Input

The input contains a number n ($1 <= n <= 10^7$) per line, and will be terminated by end-of-file.

Output

For each number n in the input list, print TERM n IS a/b where a/b is the n term in Cantor's enumeration.

Sample input	Sample Output
3 14 7	TERM 3 IS 2/1 TERM 14 IS 2/4 TERM 7 IS 1/4

Problem 4.

RSA

Source file name: rsa.c, rsa.cpp or rsa.java
Input: rsa.in
Output: standar output

One of the methods of cryptography used at present, is algorithm RSA, this system is based on the difficulty of the mathematical problem of the factoring of a compound number, for what algorithms of time do not exist polinomial, and in the facility of the inverse operation, to multiply it.

The basic functioning becomes considering the variables p, q, e and M where M is the Message that must enter and change to a Hexadecimal output.
The conditions to validate the information are:

- p and q must be prime numbers.
- The message must not have a limit of information and / or characters.
- The exit must be only hexadecimal.
- n = p * q
- Φ: Φ (n)=(p-1)(q-1)
- Find a number e that the MCD(e, Φ (n))=1.
- You must calculate d=((Y* Φ(n))+1)/e for Y=1,2,3,... until d becomes integer.
- Numbers (e, d) are the public key
- Numbers (d, n) are the private key
- The encrypted message is obtained by the formula: C = (M^e) mod n
- The original message is obtained by the formula: M = (C^d) mod n

Input

The entry consists of a several cases. Each one of cases has the number p, number q, number e and the message M.
The coding will be realized applying the formula: C = (M^e) mod n

Output

The output is determined by the encrypted message coded in Hexadecimal.

Sample input	Sample Output
53 23 15 Hello 37 19 5 RSA TEST	01EF00A703FD03FD00CC 02B100B6017500F2022000E900D200A5

Problem 5.

Climbing Worm

Source file name: worm.c, worm.cpp or worm.java
Input: worm.in
Output: standar output

An inch worm is at the bottom of well n inches deep. It has enough energy to climb u inches every minute, but then has to rest a minute before climbing again. During the rest, it slips down d inches. The process of climbing and resting then repeats. How long before the worm climbs out of the well? We'll always count a portion of a minute as a whole minute and if the worm just reaches the top of the well at the end of its climbing, we'll assume the worm makes it out.

Input

There will be multiple problem instances. Each line will contain 3 positive integers n, u and d. These give the values mentioned in the paragraph above. Furthermore, you may assume $d < u$ and $n < 100$. A value of $n = 0$ indicates end of input.

Output

Each input instance should generate a single integer on a line, indicating the number of minutes it takes for the worm to climb out of the well.

Sample input	Sample Output
10 2 1 20 3 1 0 0 0	17 19

Problem 6.

Symmetric Order

Source file name: order.c, order.cpp or order.java
Input: order.in
Output: standar output

In your job at Albatross Circus Management (yes, it's run by a bunch of clowns), you have just finished writing a program whose output is a list of names in nondescending order by length (so that each name is at least as long as the one preceding it). However, your boss does not like the way the output looks, and instead wants the output to appear more symmetric, with the shorter strings at the top and bottom and the longer strings in the middle. His rule is that each pair of names belongs on opposite ends of the list, and the first name in the pair is always in the top part of the list. In the first example set below, Bo and Pat are the first pair, Jean and Kevin the second pair, etc.

Input

The input consists of one or more sets of strings, followed by a final line containing only the value 0. Each set starts with a line containing an integer, n, which is the number of strings in the set, followed by n strings, one per line, sorted in nondescending order by length. None of the strings contain spaces. There is at least one and no more than 15 strings per set. Each string is at most 25 characters long.

Output

For each input set print "SET n" on a line, where n starts at 1, followed by the output set as shown in the sample output.

Sample input	Sample Output
7 Bo Pat Jean Kevin Claude William Marybeth 6 Jim Ben Zoe Joey Frederick Annabelle 0	SET 1 Bo Jean Claude Marybeth William Kevin Pat SET 2 Jim Zoe Frederick Annabelle Joey Ben