HKUST Programming Contest 2007 Fall

Date: October 7th, 2007
Time: 13:00 – 17:00
Venue: CS Lab 3

Rules:

1. Teams are ranked according to the most problems solved. Teams who solve the same number of problems are ranked by least total time. The total time is the sum of the time consumed for each problem solved. The time consumed for a solved problem is the time elapsed from the beginning of the contest to the submission of the accepted run plus 20 penalty minutes for every rejected run for that problem regardless of submission time. There is no time consumed for a problem that is not solved.
2. The correctness of each submission is judged by inputting test cases into the submitted program. The submission is regarded as correct if its output matches completely with the judge's output. The submission is judged as correct or wrong. No partial credit is given.
3. The contestants can re-submit another source code after previous wrong submissions.
4. The contestants should read the input from standard input and print the output to standard output for all the problems.
5. The time limits for all the problems are 5 seconds.
6. The programming language to be used in this contest is C/C++. The contestants use PC^2 to submit their source codes to the judge and the source codes are compiled by Visual Studio 6.0.
7. The contestants are allowed to bring any hard copies of books, notes, references, dictionaries and sketch papers to the contest site. Electronic devices are forbidden.
8. This is an individual-based contest. Contestants should follow rules and no cheating/collaboration behavior is allowed. Any offence of this rule will be punished according to the university academic regulations.

Problem A – Wild Numbers

A wild number is a string containing digits and question marks (like 36?1?8). A number X matches a wild number W if they have the same length, and every non-question mark character in W is equal to the character in the same position in X (it means that you can replace a question mark with any digit). For example, 365198 matches the wild number 36?1?8, but 360199, 361028, or 36128 does not. Write a program that reads a wild number W and a number X from input, both of length n, and determines the number of n-digit numbers that match W and are greater than X.

Input

There are multiple test cases in the input. Each test case consists of two lines of the same length. The first line contains a wild number *W*, and the second line contains an integer number *X*. *X* contains no leading 0. The length of input lines is between 1 and 10 characters. The last line of input contains a single character #.

Output

For each test case, write a single line containing the number of *n*-digit numbers matching *W* and greater than *X* (*n* is the length of *W* and *X*).

Sample Input

```
36?1?8
236428
8?3
910
?
5
#
```

Sample Output

```
100
0
4
```

Problem B – The Idiot

There is just one basic rule in the Idiot of the Year Contest (IYC)! The contestant picks a random digit between 0 and 9, computes the factorial of the day of the year he/she is born, and counts the how many times the digit picked appears in the factorial. The contestant with highest count is the Idiot of the Year! For example, if you are born on 5^{th} of Mordad which is the 129^{th} day of the year, and you pick the digit 6, your score will be the number of times the digit 6 appears in 129! (that is $1\times2\times3\times...\times129$).

The chief judge of IYC wants you to write a program to get an integer which is the day of the year a contestant is born on and a digit and report the number of times the digit appears in the factorial of the first number.

Input

The first line of the input contains a single integer T which is the number of test cases, followed by T lines each containing the data for a test case having two numbers. The first number is the day of the year a contestant is born and the second one is the digit he/she has picked.

Output

The output contains T lines, each having one integer which is the number of times the digit appears in the factorial of the first number.

Sample Input

```
2
5 2
7 0
```

Sample Output

```
1
2
```

Problem C – Power Calculus

Starting with x and repeatedly multiplying by x, we can compute x^{31} with thirty multiplications:

$x^2 = x \times x$, $x^3 = x^2 \times x$, $x^4 = x^3 \times x$, ... , $x^{31} = x^{30} \times x$.

The operation of squaring can appreciably shorten the sequence of multiplications. The following is a way to compute x^{31} with eight multiplications:

$x^2 = x \times x$, $x^3 = x^2 \times x$, $x^6 = x^3 \times x^3$, $x^7 = x^6 \times x$, $x^{14} = x^7 \times x^7$, $x^{15} = x^{14} \times x$, $x^{30} = x^{15} \times x^{15}$, $x^{31} = x^{30} \times x$.

This is not the shortest sequence of multiplications to compute x^{31}. There are many ways with only seven multiplications. The following is one of them:

$x^2 = x \times x$, $x^4 = x^2 \times x^2$, $x^8 = x^4 \times x^4$, $x^{10} = x^8 \times x^2$, $x^{20} = x^{10} \times x^{10}$, $x^{30} = x^{20} \times x^{10}$, $x^{31} = x^{30} \times x$.

There however is no way to compute x^{31} with fewer multiplications. Thus this is one of the most efficient ways to compute x^{31} only by multiplications.

If division is also available, we can find a shorter sequence of operations. It is possible to compute x^{31} with six operations (five multiplications and one division):

$x^2 = x \times x$, $x^4 = x^2 \times x^2$, $x^8 = x^4 \times x^4$, $x^{16} = x^8 \times x^8$, $x^{32} = x^{16} \times x^{16}$, $x^{31} = x^{32} \div x$.

This is one of the most efficient ways to compute x^{31} if a division is as fast as a multiplication.

Your mission is to write a program to find the least number of operations to compute x^n by multiplication and division starting with x for the given positive integer n. Products and quotients appearing in the sequence of operations should be x to a positive integer's power. In other words, x^{-3}, for example, should never appear.

Input

The input is a sequence of one or more lines each containing a single integer n. n is positive and less than or equal to 1000. The end of the input is indicated by a zero.

Output

Your program should print the least total number of multiplications and divisions required to compute x^n starting with x for the integer n. The numbers should be written each in a separate line without any superfluous characters such as leading or trailing spaces.

Sample Input

```
1
31
70
91
473
512
811
953
0
```

Sample Output

```
0
6
8
9
11
9
13
12
```

Problem D – Integer Equation

Given, n, count the number of solutions to the equation x+2y+2z=n, where x, y, z, n are non negative integers.

Input
There are at most 1500 inputs. Each input is n (n<1000001) on a single line.

Output
For each input, output the number of solutions on a single line.

Sample Input
```
2
3
```

Sample Output
```
3
3
```

Problem E – Cyber Cube

Martin and Roger awaken in a maze of mysterious rooms (Don't ask the author - why). Each room has one door. The door is locked, but fortunately they have some keys (labeled with numbers) in their hand. Soon they discover some more interesting things; it seems that the basic laws of physics don't apply in this world. The world has following rules:

1. Each room is labeled by a number (1 to 2147483648)
2. Each door can be opened by any key.
3. There have two kinds of keys: Gold and Silver.
   i. Gold key opens a door to connect to a room whose number is the multiple of current room number and key number.
   ii. Silver key opens a door to connect to a room whose number is the addition of current room number and key number.
4. After they open the door and walk into the room behind the door, the door will be closed automatically. They may not be able to return to previous room if they open this door again (according to rule 3).
5. If they go into a room whose number exceeds 2147483648, they will be trapped and won't be able to escape from that room.

In the beginning, they fall into **different** rooms. Martin found 9 keys labeled from 1 to 9 and all are gold keys, while poor Roger could only find a key labeled with one and it is silver. You may assume Martin always falls to room number one. In each time unit, both of them must open the door in his room using one key and walk through the door to get into the room determined by the rules in this world. Martin and Roger are very hurry to find each other. (Don't ask why) Could you help to write a program to tell whether they can meet in this maze and how much time is needed?

Input

The first line of input contains a single positive integer t($t \leq 3000$), the number of test cases. Each test case contains a single integer n ($2 \leq n \leq 2147483648$), which is the initial room number of Roger resisted.

Output

For each test case, output one line containing a single integer that holds the minimum unit of time for Martin and Roger to meet. If it is impossible, print a string "IMPOSSIBLE" instead.

Sample Input

```
5
12
162
100000
2000000000
2147483640
```

Sample Output

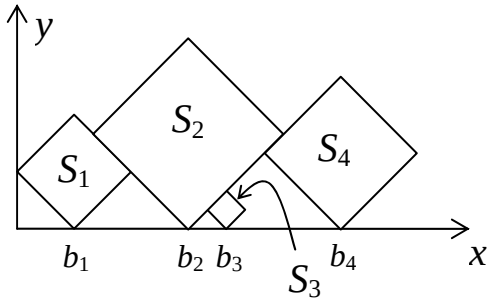
```
2
6
352
376000
IMPOSSIBLE
```

Problem F – Squares

In this problem, you are given a sequence S_1, S_2, ..., S_n of squares of different sizes. The sides of the squares are integer numbers. We locate the squares on the positive x-y quarter of the plane, such that their sides make 45 degrees with x and y axes, and one of their vertices are on y=0 line. Let b_i be the x coordinates of the bottom vertex of S_i. First, put S_1 such that its left vertex lies on x=0. Then, put S_i, (i > 1) at minimum b_i such that

a) $b_i > b_{i-1}$ and

b) the interior of S_i does not have intersection with the interior of S_1...S_{i-1}.



The goal is to find which squares are visible, either entirely or partially, when viewed from above. In the example above, the squares S_1, S_2, and S_4 have this property. More formally, S_i is visible from above if it contains a point p, such that no square other than S_i intersect the vertical half-line drawn from p upwards.

Input

The input consists of multiple test cases. The first line of each test case is n ($1 \le n \le 50$), the number of squares. The second line contains n integers between 1 to 30, where the i^{th} number is the length of the sides of S_i. The input is terminated by a line containing a zero number.

Output

For each test case, output a single line containing the index of the visible squares in the input sequence, in ascending order, separated by a blank character.

Sample Input

```
4
3 5 1 4
3
2 1 2
0
```

Sample Output

```
1 2 4
1 3
```