|  |  |
| --- | --- |
| **Palindrome**  **Solved**  Problem code: DPC206 | * [SUBMIT](https://www.codechef.com/submit/DPC206) * [MY SUBMISSIONS](https://www.codechef.com/status/DPC206,nacho0monllor) * [ALL SUBMISSIONS](https://www.codechef.com/status/DPC206) |

**All submissions for this problem are available.**

**Palindrome**

          A simple recursive method to generate a numeric palindrome from any number is to reverse its digits and add it to the original. If the sum is not a palindrome (which means, it is not the same number from left to right and right to left), repeat this procedure. For example for 195:

    195 + 951 = 786

    786 + 687 = 1473

    1473 + 3741 = 5214

    5214 + 4125 = 9339 Resulting palindrome

 In this particular case the palindrome 9339 appeared after 4th addition. This method leads to palindromes in a few step for almost all of the integers. But there are interesting exceptions. 196 is the first number for which no palindrome has been found. It is not proven though, that there is no such a palindrome.

 Your task is to write a program that gives the resulting palindrome and the number of iterations (additions) to compute it. All tests data in this problem will have an answer, will be computable with less than 1000 iterations (additions), which will yield a palindrome that is not greater than 4,294,967,295.

**Input**

                    The first line will have a number N (0<N<=100) with the number of test cases, the next N lines will each have a number P to compute its palindrome.

**Output**

          For each of the N numbers you will have to write a line with the minimum number of iterations (additions) to get to the palindrome and the resulting palindrome separated by one space.

**Sample Input**

3

195

265

750

**Sample Output**

4 9339

5 45254

3 6666

<https://www.codechef.com/problems/DPC206>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static bool esPalin(string s)

{

char[] rev = s.ToCharArray();

Array.Reverse(rev);

if (new string(rev) == s)

{

return true;

}

return false;

}

static string rev(string s)

{

char[] r = s.ToCharArray();

Array.Reverse(r);

return new string(r);

}

static void Main(string[] args)

{

int t = int.Parse(Console.ReadLine());

while (t-- > 0)

{

int p = int.Parse(Console.ReadLine());

int res = p;

int adiciones = 0;

while (!esPalin(res.ToString()))

{

// Console.WriteLine(p + " " + rev(res.ToString()));

res = p + int.Parse(rev(p.ToString()));

adiciones++;

p = res;

}

Console.WriteLine(adiciones + " " + res);

}

Console.ReadLine();

}

}

}