1024. Palindromic Number (25)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

A number that will be the same when it is written forwards or backwards is known as a Palindromic Number. For example, 1234321 is a palindromic number. All single digit numbers are palindromic numbers.

Non-palindromic numbers can be paired with palindromic ones via a series of operations. First, the non-palindromic number is reversed and the result is added to the original number. If the result is not a palindromic number, this is repeated until it gives a palindromic number. For example, if we start from 67, we can obtain a palindromic number in 2 steps: 67 + 76 = 143, and 143 + 341 = 484.

Given any positive integer N, you are supposed to find its paired palindromic number and the number of steps taken to find it.

**Input Specification:**

Each input file contains one test case. Each case consists of two positive numbers N and K, where N (<= 1010) is the initial numer and K (<= 100) is the maximum number of steps. The numbers are separated by a space.

**Output Specification:**

For each test case, output two numbers, one in each line. The first number is the paired palindromic number of N, and the second number is the number of steps taken to find the palindromic number. If the palindromic number is not found after K steps, just output the number obtained at the Kth step and K instead.

**Sample Input 1:**

67 3

**Sample Output 1:**

484

2

**Sample Input 2:**

69 3

**Sample Output 2:**

1353

3

#include<iostream>

#include<string>

#include<algorithm>

#include<queue>

#include<vector>

#include<sstream>

#define ll long long

#include<stack>

using namespace std;

string &reverse(string num)

{

string res=num;

for (int i = 0; i < res.size() / 2; i++)

{

swap(res[i], res[res.size() - 1 - i]);

}

return res;

}

int check(string num)

{

if (num == reverse(num))

return 1;

else

return 0;

}

string& addition(string num)

{

string rev = reverse(num);

string res;

res.resize(rev.size() + 1);

int former = 0;

for (int i = 0; i < rev.size(); i++)

{

int t\_a = rev[i] - '0', t\_b = num[i] - '0';

if (t\_a + t\_b + former>9)

{

res[i] = t\_a + t\_b + former - 10 + '0';

former = 1;

}

else

{

res[i] = t\_a + t\_b + former+'0';

former = 0;

}

}

if (former)

res[rev.size()] = '1';

return reverse(res);

}

int main()

{

string num;

int time;

cin >> num >> time;

for (ll i = 0; i < time; i++)

{

if (check(num))

{

cout << num << endl;

cout << i;

return 0;

}

num=addition(num);

}

cout << num << endl << time;

}