ReportTermProject Algorithm Design(CS3201) Section541 Semester 2/2016

By **Artisd Chanyawadee 5712266**

Chawan Vattanalap 5737444

PALIN - The Next Palindrome

Problem

A positive integer is called a palindrome if its representation in the decimal system is the same when read from left to right and from right to left. For a given positive integer K of not more than 1000000 digits, write the value of the smallest palindrome larger than K to output. Numbers are always displayed without leading zeros.

Input

The first line contains integer t, the number of test cases. Integer K are given in the next t lines

Output

For each K, output the smallest palindrome larger than K.

Sample

Input	Output
12 0 9 808 2133 199 1999 319887788993 333321 111 1111 999 9999 99999	1 11 818 2222 202 2002 319888888913 333333 121 1221 909 10001

ProblemAuthor: adrian
ProblemSource: SPOJ
Concept Difficulty: 38%
Timelimit:2-9 second
Memory limit: 1536MB

Solution

Take the input as array of character. Then there will be 2 big cases.

1. **The in put number is palindrome and has all 9s. For example “999”. Output should be “1001”.**

```
bool checkIfAll9s(string num){
    for(int i=0; i<num.length(); i++){
        if(num[i] != '9'){
            return false;
        }
    }
    return true;
}
```

2. **In other cases, we create a creatLeftMirror function.**

```
//Function to check if number in string1 is greater than number in string2
int ifGreater(string num1,string num2){
    for(int i=0; i<num1.length(); i++){
        int digit1 = num1[i] - '0';
        int digit2 = num2[i] - '0';
        //If number 2 is greater
        if(digit1 < digit2){
            return 1;
        }
        //If number 1 is greater
        else if(digit1 > digit2){
            return -1;
        }
    }
    //If all the digits were equal
    return 0;
}
```

And we compare the last digit in the left half and the first digit in the right half by sending them to ifGreater function.

```
string createLeftMirrored(string num){
    int len = num.length();
    //If number is of even length
    if(len % 2 == 0){
        string left_half = num.substr(0,len/2);
        //right_half is just reverse of left_half
        string right_half = left_half;
        reverse(right_half.begin(),right_half.end());
        return left_half + right_half;
    }
    //If number is of odd length
    else{
        string left_half = num.substr(0,len/2);
        //right_half is just reverse of left_half
        string right_half = left_half;
        reverse(right_half.begin(),right_half.end());
        return left_half + num[len/2] + right_half;
    }
}
```

If the leftMirrored is greater than the input, then we have the next palindrome by using generateNextPalindrome function.

```
string generateNextPalindrome(string num){
    int len = num.length();
    //In case of even number of digits
    if(len % 2 == 0){
        int center_right = len/2;
        int center_left = center_right - 1;
        while(num[center_left] == '9' && num[center_right] == '9'){
            num[center_left] = '0';
            num[center_right] = '0';
            center_left -= 1;
            center_right += 1;
        }
        num[center_left] += 1;
        num[center_right] += 1;
    }
```

```
//In case of odd number of digits
else{
    int dead_center = len/2;
    if(num[dead_center] != '9'){
        num[dead_center] += 1;
    }
    else{
        num[dead_center] = '0';
        int center_left = dead_center - 1;
        int center_right = dead_center + 1;
        while(num[center_left] == '9' && num[center_right] == '9'){
            num[center_left] = '0';
            num[center_right] = '0';
            center_left = center_left - 1;
            center_right = center_right + 1;
        }
        num[center_left] += 1;
        num[center_right] += 1;
    }
}
```

Code

```
#include <iostream>
#include <string>
#include <algorithm>

using namespace std;
string createLeftMirrored(string num){ int len =
        num.length();
        //If number is of even length if(len % 2 == 0){
                string left_half = num.substr(0,len/2);
                //right_half is just reverse of left_half string right_half = left_half;
                reverse(right_half.begin(),right_half.end()); return left_half +
                right_half;
        }
        //If number is of odd length else{
                string left_half = num.substr(0,len/2);
                //right_half is just reverse of left_half string right_half = left_half;
                reverse(right_half.begin(),right_half.end()); return left_half +
                num[len/2] + right_half;
        }
}

//Function to check if number in string1 is greater than number in  string2
int ifGreater(string num1,string num2){ for(int i=0;
        i<num1.length(); i++){
```

```
int digit1 = num1[i] - '0'; int digit2 = num2[i] - '0';
                    //If number 2 is greater if(digit1 <
                    digit2){
                            return 1;
                    }
                    //If number 1 is greater else if(digit1 >
                    digit2){
                            return -1;
                    }
            }
            //If all the digits were equal return 0;
    }

    string generateNextPalindrome(string num){ int len =
            num.length();
            //In case of even number of digits if(len % 2 == 0){
                    int center_right = len/2;
                    int center_left = center_right - 1;
                    while(num[center_left] == '9' && num[center_right] == '9'){ num[center_left] = '0';
                            num[center_right] = '0';
                            center_left -= 1;
                            center_right += 1;
                    }
                    num[center_left] += 1;
                    num[center_right] += 1;
            }
            //In case of odd number of digits else{
                    int dead_center = len/2; if(num[dead_center] !=
                    '9'){
                            num[dead_center] += 1;
                    }
                    else{
                            num[dead_center] = '0';
                            int center_left = dead_center - 1; int center_right =
                            dead_center + 1;
                            while(num[center_left] == '9' && num[center_right] ==
    '9'){
                                    num[center_left] = '0';
                                    num[center_right] = '0'; center_left = center_left -
                                    1; center_right = center_right + 1;
                            }
                            num[center_left] += 1;
                            num[center_right] += 1;
                    }
            }
            return num;
    }
```

```
bool checkIfAll9s(string num){
	for(int i=0; i<num.length(); i++){ if(num[i] != '9'){
			return false;
		}
	}
	return true;
}

int main(){
	//For Fast IO ios_base::sync_with_stdio(false); cin.tie(NULL);

int main(){
	int t; string num;
	cin>>t;
	for(int testcase=0; testcase<t; testcase++){ cin>>num;
		if(checkIfAll9s(num) == true){
			//For 999, answer will be 1001, For 9999, answer is
10001
			//1 in the start
			string nextPalindrome = "1";
			//1 followed by n-1 zeros, where n is length of the
number
			for(int i=0; i<num.length()-1; i++){ nextPalindrome
				+= '0';
			}
			//Last digit is again 1 nextPalindrome
			+= '1'; cout<<nextPalindrome<<"\n";

		 }
		   else{ string leftMirrored = createLeftMirrored(num);
			//If the leftMirrored is greater than the num, then we
have the next palindrome
			if(ifGreater(num,leftMirrored) == 1){
				cout<<leftMirrored<<"\n";
			 }
			   else{
				string nextPalindrome =
generateNextPalindrome(leftMirrored);
				//Keep incrementing the palindrome until it is
greater than num
				while(ifGreater(num,nextPalindrome) != 1){ nextPalindrome
					=
generateNextPalindrome(nextPalindrome);
				}
				cout<<nextPalindrome<<"\n";
			}
		}
	}
	return 0
```