Week 5 – 02:

ROLL NO.:240801030

Name: Arumugam.k

Q1) The k-digit number N is an Armstrong number if and only if the k-th power of each digit sums to N.

Given a positive integer N, return true if and only if it is an Armstrong number.

Note: 1 <= N <= 10^8

Hint: 153 is a 3-digit number, and 153 = 1^3 + 5^3 + 3^3.

Sample Input:

153

Sample Output:

true

Sample Input:

123

Sample Output:

false

Sample Input:

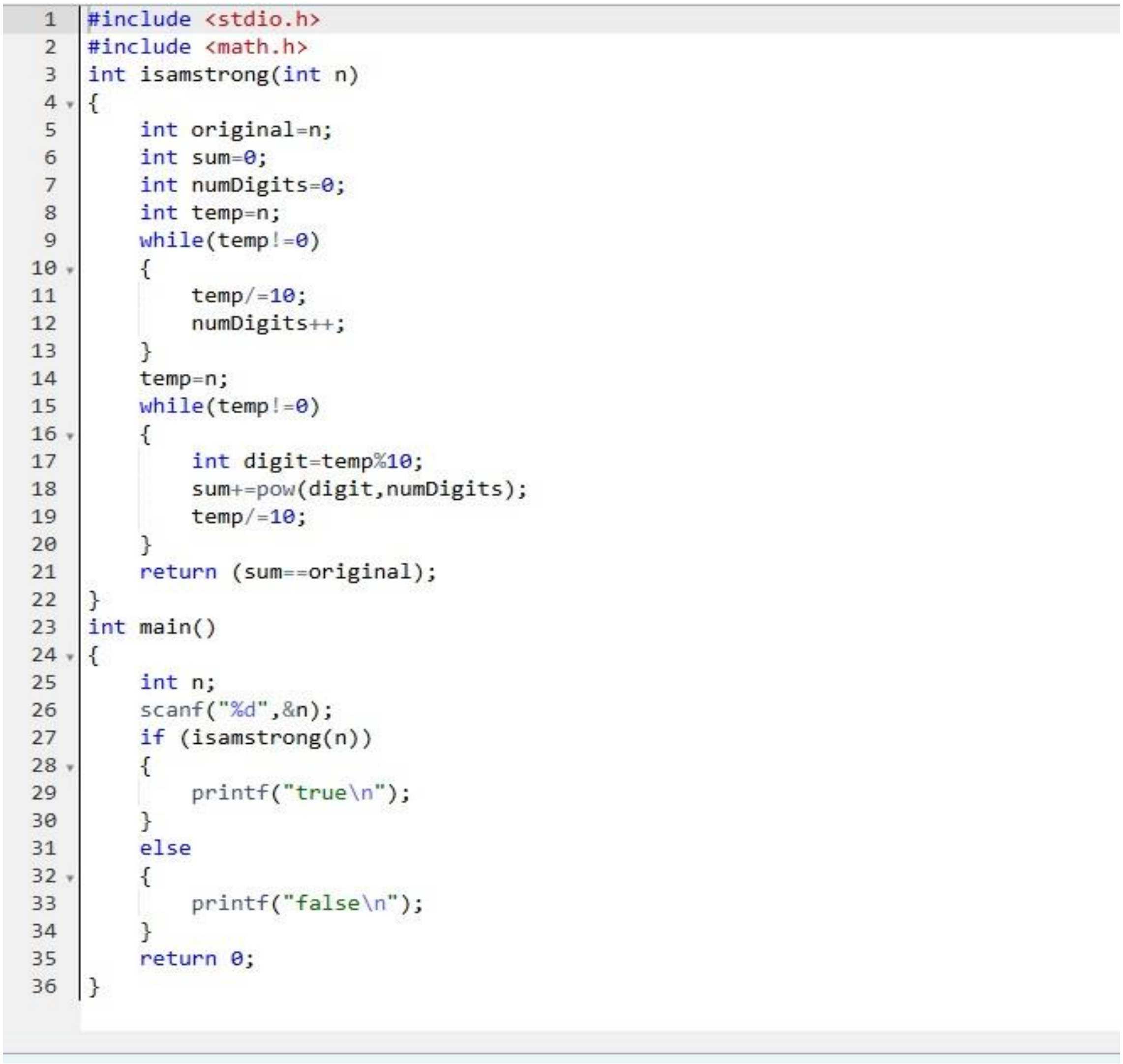
1634

Sample Output:

True

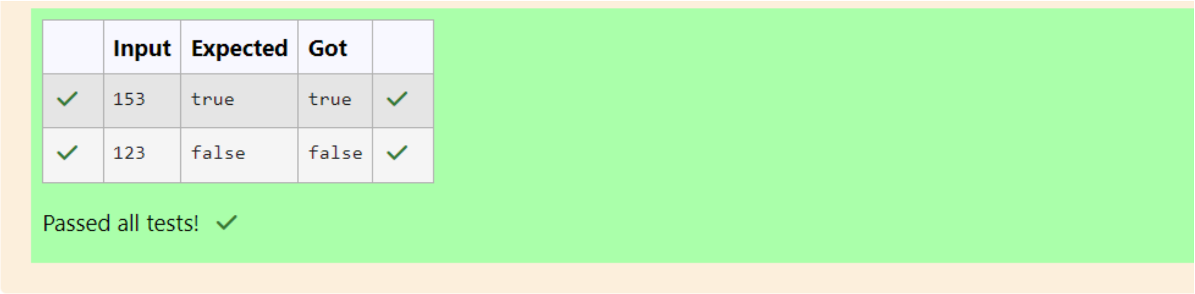
CODE:

| | |
|---|---|
| **Status** | Finished |
| **Started** | Monday, 13 January 2025, 10:26 AM |
| **Completed** | Monday, 13 January 2025, 10:57 AM |
| **Duration** | 31 mins 26 secs |



```c
#include <stdio.h>
#include <math.h>
int isamstrong(int n)
{
    int original=n;
    int sum=0;
    int numDigits=0;
    int temp=n;
    while(temp!=0)
    {
        temp/=10;
        numDigits++;
    }
    temp=n;
    while(temp!=0)
    {
        int digit=temp%10;
        sum+=pow(digit,numDigits);
        temp/=10;
    }
    return (sum==original);
}
int main()
{
    int n;
    scanf("%d",&n);
    if (isamstrong(n))
    {
        printf("true\n");
    }
    else
    {
        printf("false\n");
    }
    return 0;
}
```

OUTPUT:



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 153 | true | true | ✓ |
| ✓ | 123 | false | false | ✓ |

Passed all tests! ✓

Q2) Take a number, reverse it and add it to the original number until the obtained number is a palindrome.

Constraints

1<=num<=99999999

Sample Input 1

32

Sample Output 1

55

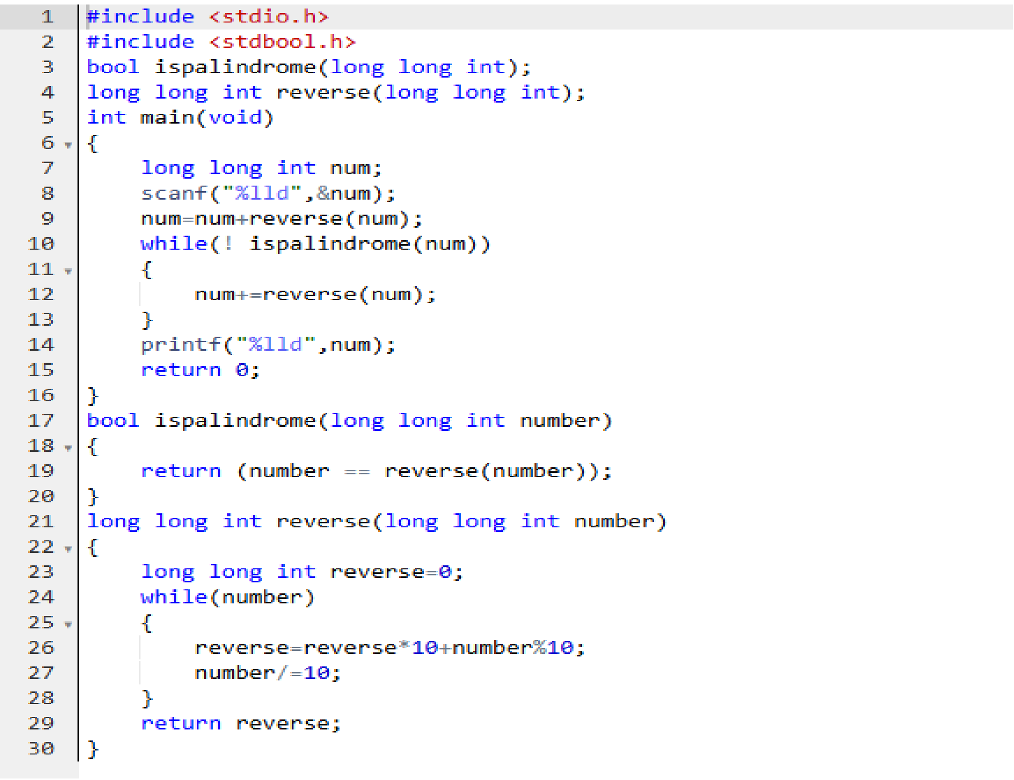
Sample Input 2

789

Sample Output 2

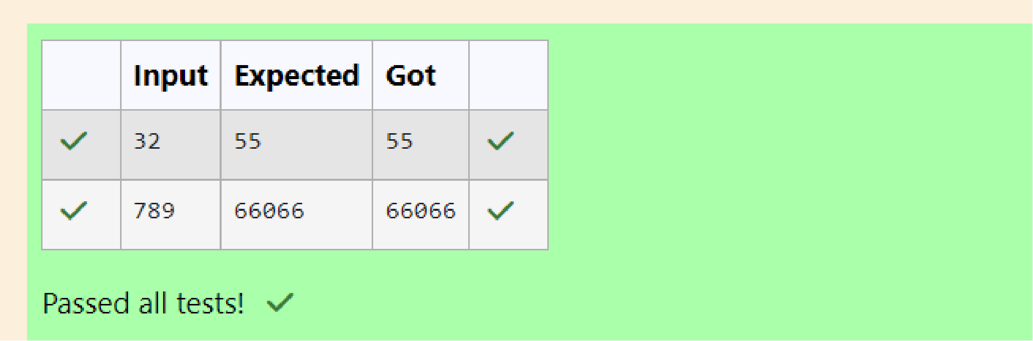
66066

CODE:



```c
#include <stdio.h>
#include <stdbool.h>
bool ispalindrome(long long int);
long long int reverse(long long int);
int main(void)
{
    long long int num;
    scanf("%lld",&num);
    num=num+reverse(num);
    while(! ispalindrome(num))
    {
        num+=reverse(num);
    }
    printf("%lld",num);
    return 0;
}
bool ispalindrome(long long int number)
{
    return (number == reverse(number));
}
long long int reverse(long long int number)
{
    long long int reverse=0;
    while(number)
    {
        reverse=reverse*10+number%10;
        number/=10;
    }
    return reverse;
}
```

OUTPUT:



|   | Input | Expected | Got |   |
|---|---|---|---|---|
| ✓ | 32 | 55 | 55 | ✓ |
| ✓ | 789 | 66066 | 66066 | ✓ |

Passed all tests! ✓

Q3) A number is considered lucky if it contains either 3 or 4 or 3 and 4 both in it. Write a program to print the nth lucky number. Example, 1st lucky number is 3, and 2nd lucky number is 4 and 3rd lucky number is 33 and 4th lucky number is 34 and so on. Note that 13, 40 etc., are not lucky as they have other numbers in it.

The program should accept a number 'n' as input and display the nth lucky number as output.

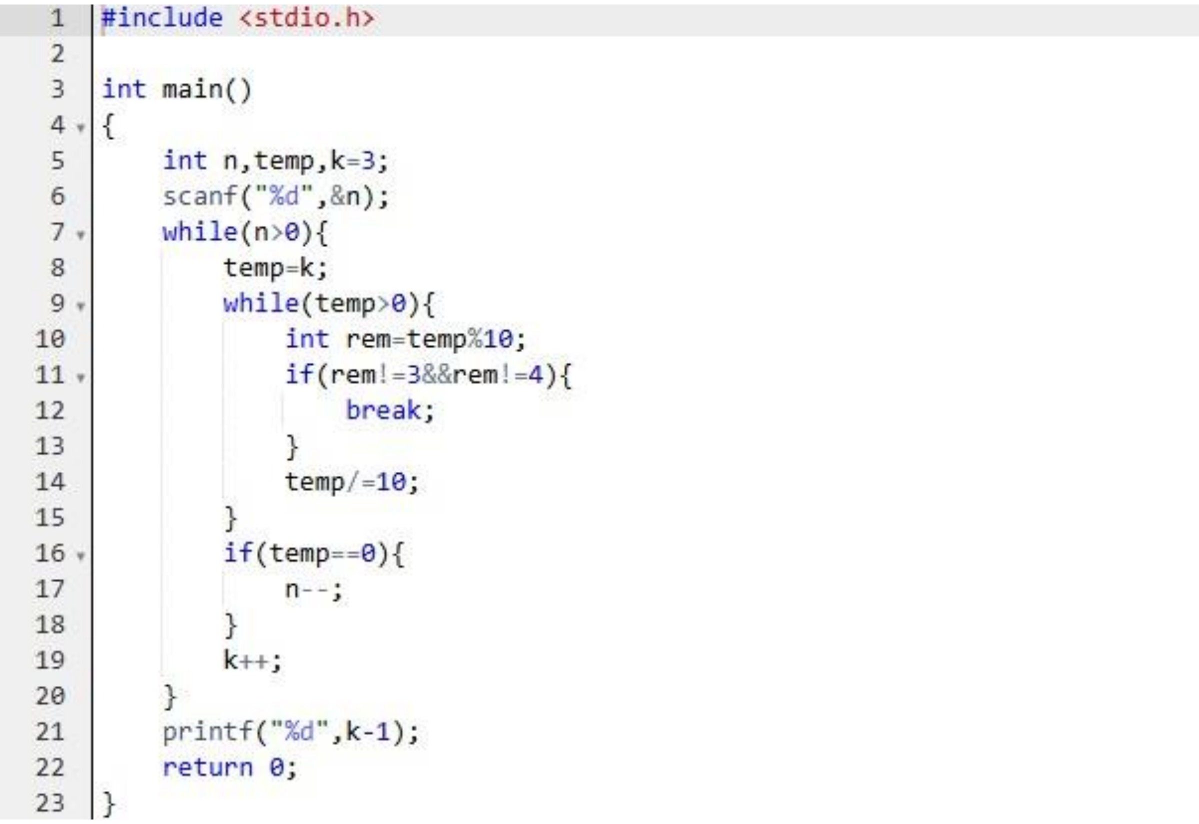
Sample Input 1:

3

Sample Output 1:

33

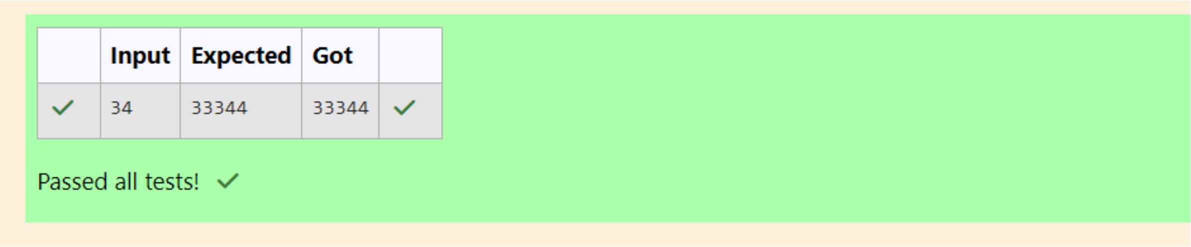
CODE:



```
#include <stdio.h>

int main()
{
    int n,temp,k=3;
    scanf("%d",&n);
    while(n>0){
        temp=k;
        while(temp>0){
            int rem=temp%10;
            if(rem!=3&&rem!=4){
                break;
            }
            temp/=10;
        }
        if(temp==0){
            n--;
        }
        k++;
    }
    printf("%d",k-1);
    return 0;
}
```

OUTPUT:



|  | Input | Expected | Got |  |
|---|---|---|---|---|
| ✓ | 34 | 33344 | 33344 | ✓ |

Passed all tests! ✓