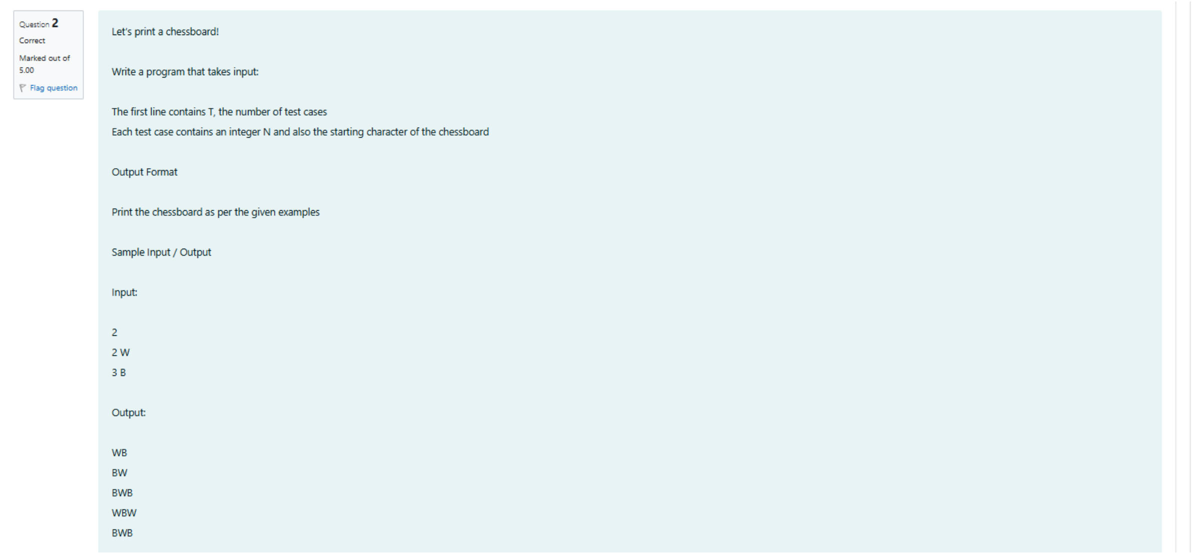


Question **2**

Correct

Marked out of 5.00

Flag question

Let's print a chessboard!

Write a program that takes input:

The first line contains T, the number of test cases
Each test case contains an integer N and also the starting character of the chessboard

Output Format

Print the chessboard as per the given examples

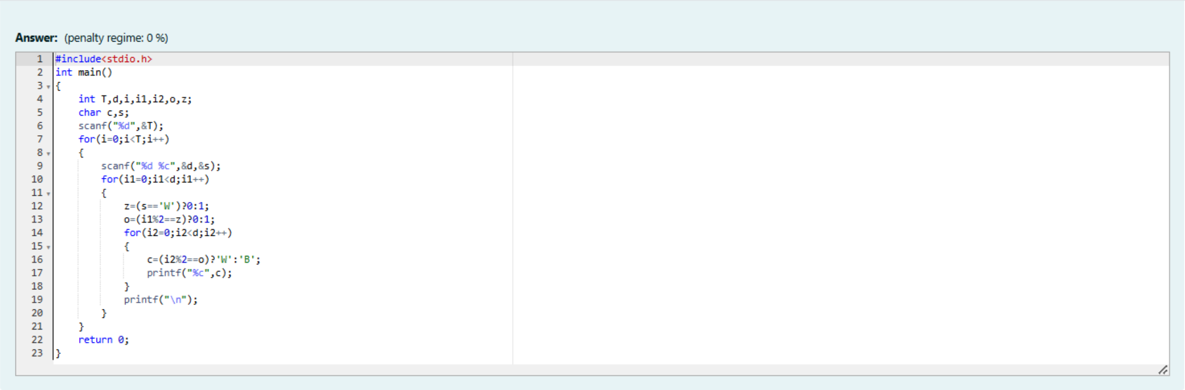
Sample Input / Output

Input:

2
2 W
3 B

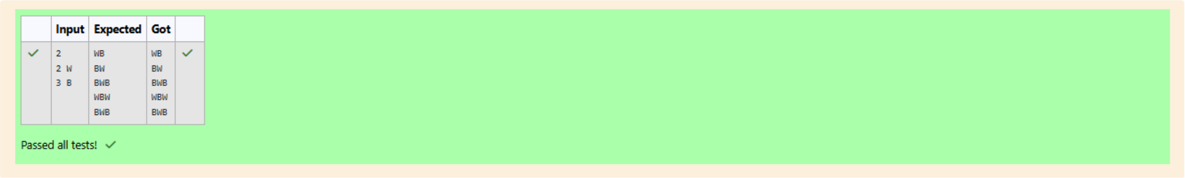
Output:

WB
BW
BWB
WBW
BWB



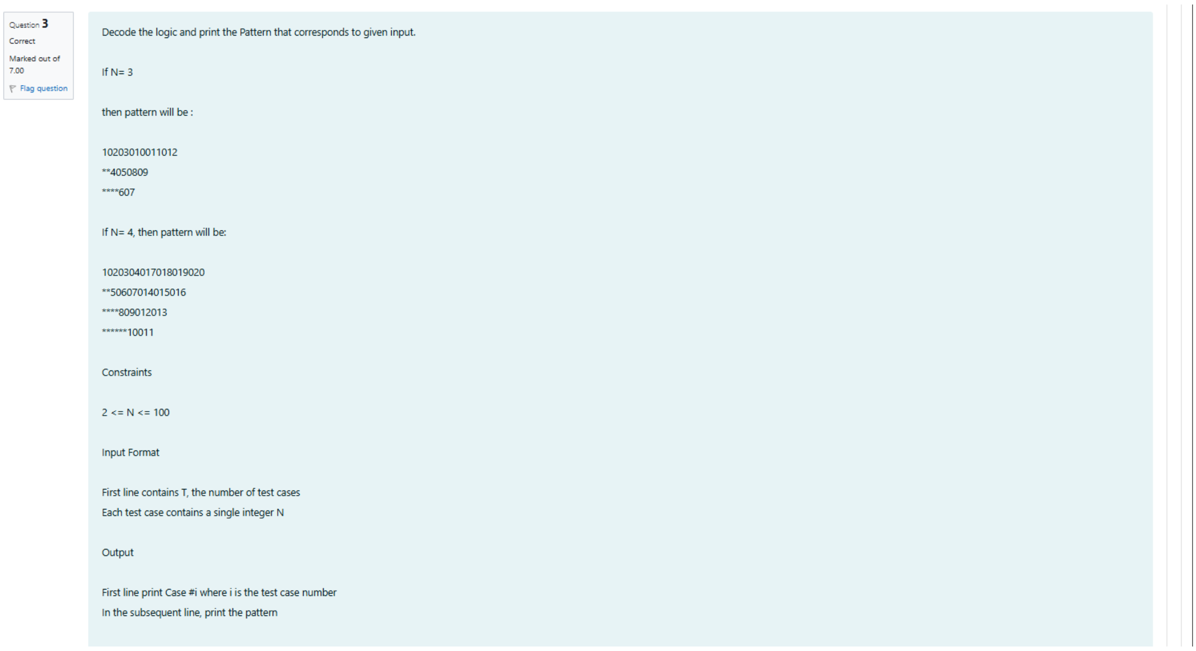
Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int T,d,i,i1,i2,o,z;
    char c,s;
    scanf("%d",&T);
    for(i=0;i<T;i++)
    {
        scanf("%d %c",&d,&s);
        for(i1=0;i1<d;i1++)
        {
            z=(s=='W')?0:1;
            o=(i1%2==z)?0:1;
            for(i2=0;i2<d;i2++)
            {
                c=(i2%2==o)?'W':'B';
                printf("%c",c);
            }
            printf("\n");
        }
    }
    return 0;
}
```



	Input	Expected	Got	
✓	2 2 W 3 B	WB BW BWB WBW BWB	WB BW BWB WBW BWB	✓

Passed all tests! ✓



Question **3**

Correct

Marked out of 7.00

Flag question

Decode the logic and print the Pattern that corresponds to given input.

If N= 3

then pattern will be :

10203010011012
**4050809
****607

If N= 4, then pattern will be:

1020304017018019020
**50607014015016
****809012013
******10011

Constraints

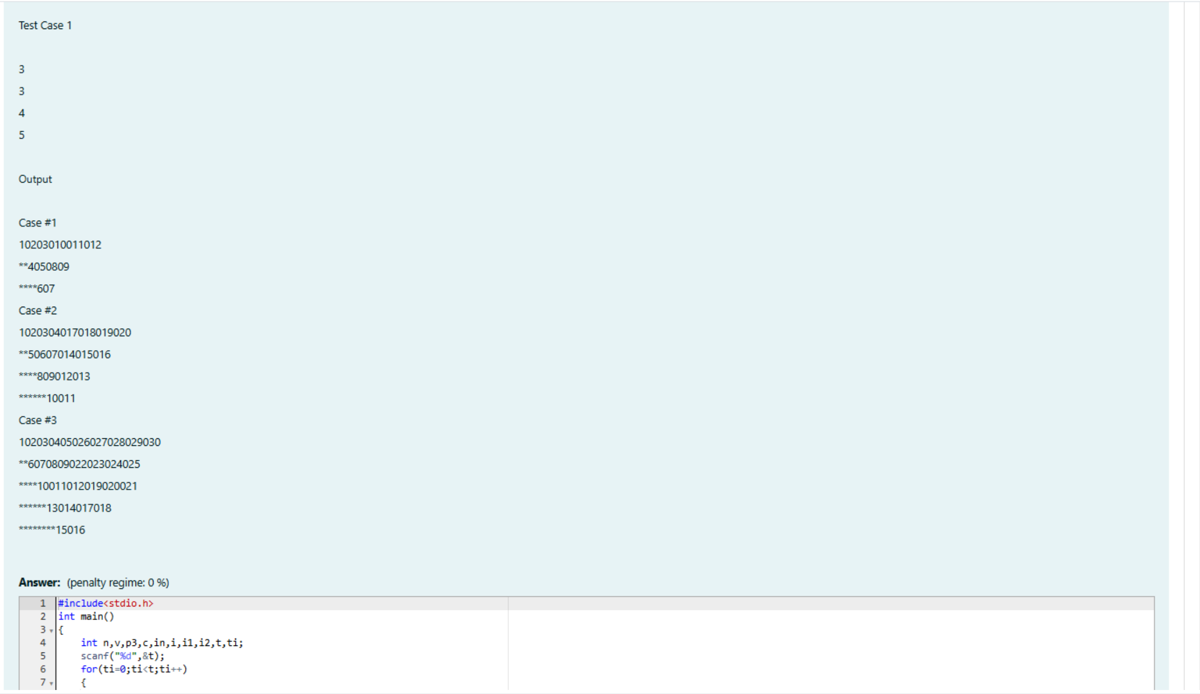
2 <= N <= 100

Input Format

First line contains T, the number of test cases
Each test case contains a single integer N

Output

First line print Case #i where i is the test case number
In the subsequent line, print the pattern



Test Case 1

3

3

4

5

Output

Case #1

10203010011012

**4050809

****607

Case #2

1020304017018019020

**50607014015016

****809012013

******10011

Case #3

102030405026027028029030

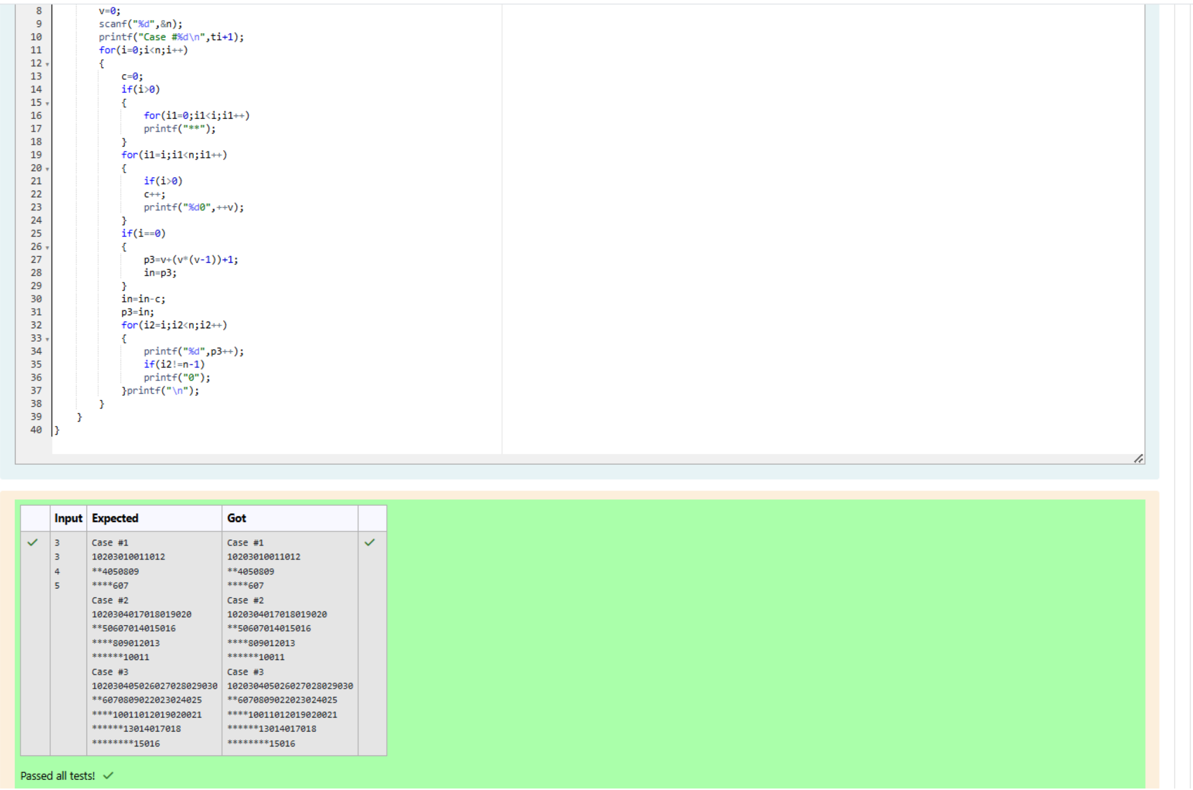
**6070809022023024025

****10011012019020021

******13014017018

********15016

Answer: (penalty regime: 0 %)

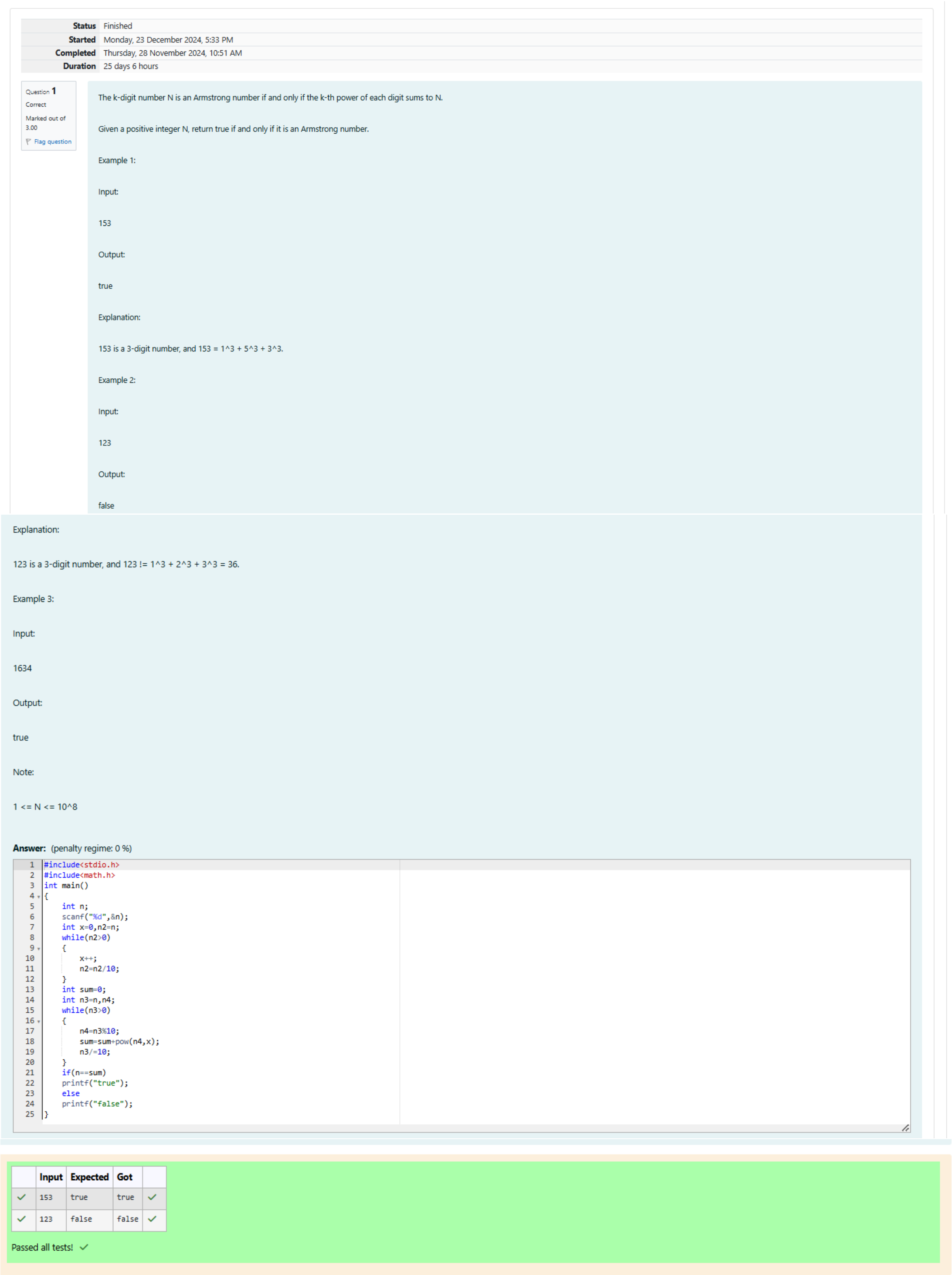


```c
#include<stdio.h>
int main()
{
    int n,v,p3,c,in,i,i1,i2,t,ti;
    scanf("%d",&t);
    for(ti=0;ti<t;ti++)
    {
        v=0;
        scanf("%d",&n);
        printf("Case #%d\n",ti+1);
        for(i=0;i<n;i++)
        {
            c=0;
            if(i>0)
            {
                for(i1=0;i1<i;i1++)
                printf("**");
            }
            for(i1=i;i1<n;i1++)
            {
                if(i>0)
                c++;
                printf("%d0",++v);
            }
            if(i==0)
            {
                p3=v+(v*(v-1))+1;
                in=p3;
            }
            in=in-c;
            p3=in;
            for(i2=i;i2<n;i2++)
            {
                printf("%d",p3++);
                if(i2!=n-1)
                printf("0");
            }printf("\n");
        }
    }
}
```

	Input	Expected	Got	
✓	3 3 4 5	Case #1 10203010011012 **4050809 ****607 Case #2 1020304017018019020 **50607014015016 ****809012013 ******10011 Case #3 102030405026027028029030 **6070809022023024025 ****10011012019020021 ******13014017018 ********15016	Case #1 10203010011012 **4050809 ****607 Case #2 1020304017018019020 **50607014015016 ****809012013 ******10011 Case #3 102030405026027028029030 **6070809022023024025 ****10011012019020021 ******13014017018 ********15016	✓

Passed all tests! ✓

WEEK 05-02



Status	Finished
Started	Monday, 23 December 2024, 5:33 PM
Completed	Thursday, 28 November 2024, 10:51 AM
Duration	25 days 6 hours

Question **1**

Correct

Marked out of 3.00

Flag question

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.

Example 1:

Input:

153

Output:

true

Explanation:

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

Example 2:

Input:

123

Output:

false

Explanation:

123 is a 3-digit number, and 123 != 1^3 + 2^3 + 3^3 = 36.

Example 3:

Input:

1634

Output:

true

Note:

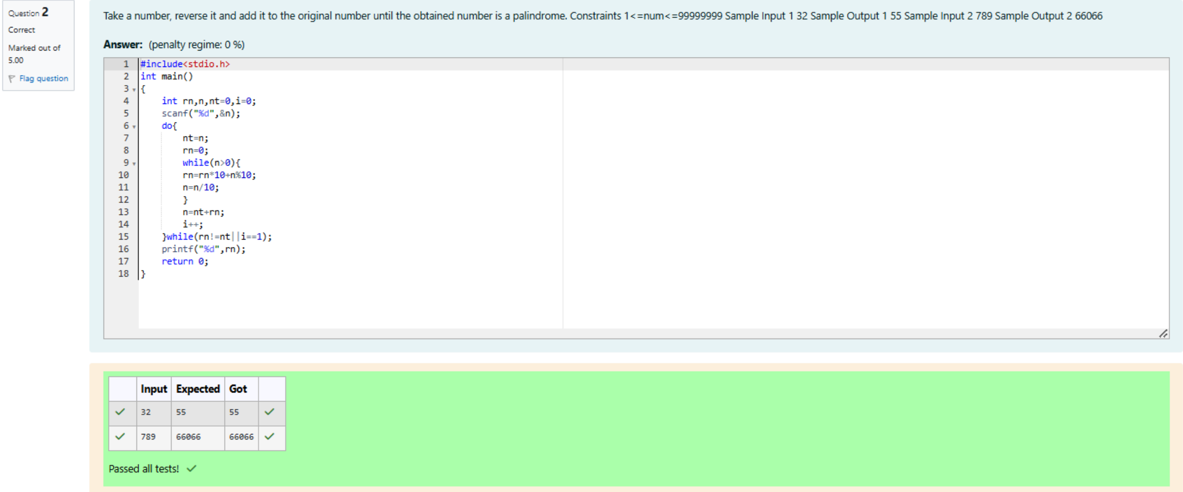
1 <= N <= 10^8

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
#include<math.h>
int main()
{
    int n;
    scanf("%d",&n);
    int x=0,n2=n;
    while(n2>0)
    {
        x++;
        n2=n2/10;
    }
    int sum=0;
    int n3=n,n4;
    while(n3>0)
    {
        n4=n3%10;
        sum=sum+pow(n4,x);
        n3/=10;
    }
    if(n==sum)
    printf("true");
    else
    printf("false");
}
```

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

Passed all tests! ✓



Question 2

Correct

Marked out of 5.00

Flag question

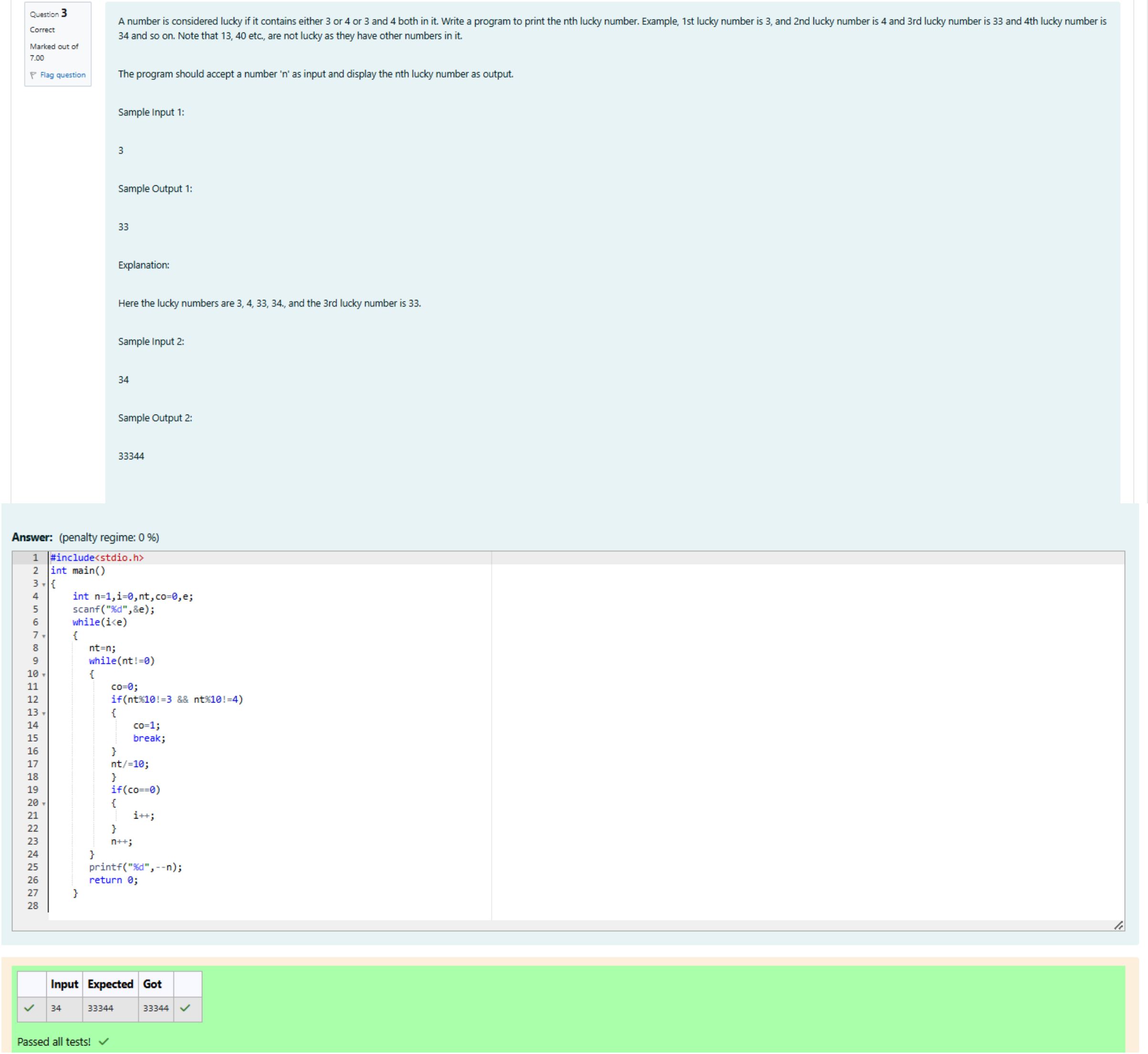
Take a number, reverse it and add it to the original number until the obtained number is a palindrome. Constraints 1<=num<=99999999 Sample Input 1 32 Sample Output 1 55 Sample Input 2 789 Sample Output 2 66066

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int rn,n,nt=0,i=0;
    scanf("%d",&n);
    do{
        nt=n;
        rn=0;
        while(n>0){
        rn=rn*10+n%10;
        n=n/10;
        }
        n=nt+rn;
        i++;
    }while(rn!=nt||i==1);
    printf("%d",rn);
    return 0;
}
```

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

Passed all tests! ✓



Question 3

Correct

Marked out of 7.00

Flag question

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

Explanation:

Here the lucky numbers are 3, 4, 33, 34., and the 3rd lucky number is 33.

Sample Input 2:

34

Sample Output 2:

33344

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int n=1,i=0,nt,co=0,e;
    scanf("%d",&e);
    while(i<e)
    {
       nt=n;
       while(nt!=0)
       {
           co=0;
           if(nt%10!=3 && nt%10!=4)
           {
               co=1;
               break;
           }
           nt/=10;
           }
           if(co==0)
           {
               i++;
           }
           n++;
       }
       printf("%d",--n);
       return 0;
    }

```

	Input	Expected	Got	
✓	34	33344	33344	✓

Passed all tests! ✓