Name: Sarvesh.R

Reg No: 240701478

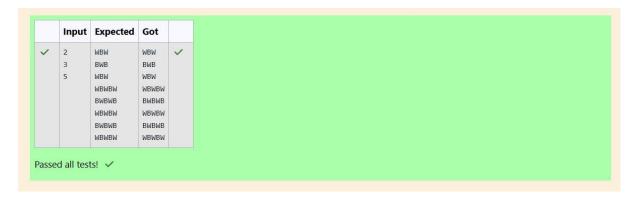
Week 5: Nested Loops - while and for, Jumps in Loops

1. single chessboard

Problem statement:

Write a program that prints a simple chessboard.	
Input format:	
The first line contains the number of inputs T. The lines after that contain a different values for size of the chessboard	
Output format:	
Print a chessboard of dimensions size * size. Print a Print W for white spaces and B for black spaces.	
Input:	
2	
3	
5	
Output:	
WBW	
BWB	
WBW	
WBWBW	
BWBWB	
WBWBW	
BWBWB	
WBWBW	

```
#include<stdio.h>
2 3 *
      int main()
      {
          int t,size;
scanf("%d",&t);
while(t--)
 4 5
 6
 7
                scanf("%d",&size);
for(int i=0;i<size;i++)</pre>
 8
10
11
                      for(int j=0;j<size;j++)</pre>
12
                          if((i+j)%2==0)
    printf("W");
else
13
14
15
                               printf("B");
16
17
                     printf("\n");
18
19
20
21
           return 0;
22 }
```

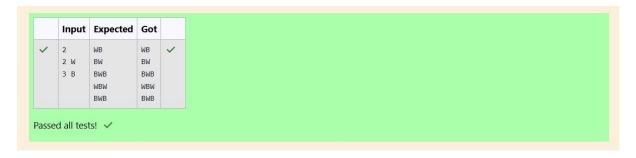


2. Print Our Own Chessboard

Problem statement:

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
Print the clessboard as per the given examples
Sample Input / Output
Input:
2
2 W
3 B
Output:
WB
BW
BWB
WBW
BWB

```
#include<stdio.h>
int main()
2 in {
            int t,n;
char ch;
scanf("%d",&t);
while(t--)
 4
 6
7
 8
                   scanf("%d %c",&n,&ch);
for(int i=0;i<n;i++)</pre>
 9
10
11 •
                         for(int j=0;j<n;j++)</pre>
12
13 1
                               if(ch=='W')
   if((i+j)%2==0)
      printf("W");
   else
14
15
16
                              printf("B");
else if((i+j)%2==0)
    printf("B");
else
17
18
19
20
21
                                     printf("W");
22
23
                         printf("\n");
24
25
26
27
             return 0;
28 }
```



3. pattern painting

Problem statement:

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

```
Case #1

10203010011012

**4050809

****607

Case #2

1020304017018019020

**50607014015016

****809012013

*******10011

Case #3

102030405026027028029030

**6070809022023024025

****10011012019020021

**********15016
```

```
1 #include<stdio.h>
     #include<string.h>
 3
     int sum(int n)
 4 * {
          return n*(n-1)/2;
 5
 6
     void BSpattern(int N)
 8 + {
          int val=0,pthree=0,cnt=0,initial;
char s[100]="**";
for(int i=0;i<N;i++)</pre>
 9
 10
 11
12 •
13
                cnt=0;
14
               if(i>0)
15 •
                    printf("%s",s);
strcat(s,"**");
16
17
18
          for(int j=i;j<N;j++)</pre>
19
20
               if(i>0)
21
22
23
                   cnt++;
24
               printf("%d",++val);
printf("0");
25
26
27
```

```
1† (1==0)
28
29
30
              int sume=sum(val)*2;
              pthree=val+sume+1;
31
              initial=pthree;
32
33
         initial=initial-cnt;
34
         pthree=initial;
35
          for(int k=i;k<N;k++)
36
37
              printf("%d",pthree++);
if(k!=N-1)
38
39
40
                   printf("0");
41
42
43
44
         printf("\n");
45
46
47
48
    int main()
49
         int N,t;
scanf("%d",%t);
for(int i=1;i<=t;i++)</pre>
50
51
52
53
              scanf("%d",&N);
printf("Case #%d\n",i);
54
55
56
              BSpattern(N);
57
         return 0;
58
59
```



4. Armstrong number

Problem statement:

```
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

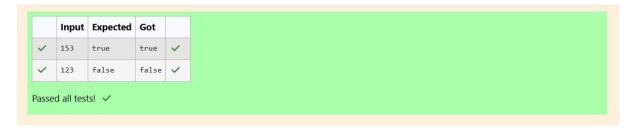
Output:

true

Explanation:

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

```
#include<stdio.h>
    #include<math.h>
    int main()
 4 * {
         int n,org,count=0,sum=0;
scanf("%d",&n);
 5
 6
         org=n;
while(n>0)
10
              count++;
11
             n/=10;
13
         n=org;
14
         while(n>0)
15
             int t=n%10;
sum+=pow(t,count);
16
17
             n/=10;
18
19
20
         if(org==sum)
21
             printf("true");
22
23
24
             printf("false");
25
26
         return 0;
27
28 }
```



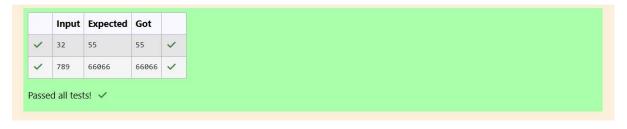
5. Reverse and Add Until Get a Palindrome

Problem statement:

Program:

```
#include<stdio.h>
 2
    int main()
 3 -
     long long int n,z,rev,temp1,temp2;
| scanf("%lld",&n);
while(1)
 4
5
 6
 8
              temp1=n,rev=0;
9
             while(n)
10
                  rev=rev*10+(n%10);
11
                  n/=10;
12
13
             z=temp1+rev;
temp2=z,rev=0;
14
15
             while(z)
16
17
18
                rev=rev*10+(z%10);
19
               z=z/10;
20
21
              if(temp2==rev){
22
                  break;
23
24
             n=temp2;
25
         printf("%lld",temp2);
26
27
         return 0;
28
```

Test cases:



6. Lucky number

Problem statement:

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

```
#include<stdio.h>
 int islucky(int num){
while(num>0)
 4 *
               int digit = num%10;
if(digit!=3&&digit!=4){
   return 0;
 5
 6 <sub>*</sub>
 8
               }num/=10;
10
          return 1;
11
int findnthlucky(int n)
13 v {
14
          int count=0,num=1;
          while(1)
15
16 *
17
               if(islucky(num))
18 *
19
                    count++;
                    if(count==n){
    return num;
20 •
21
22
23
24
               num++;
25
26
    }
int main()
27
28 * {
          int n;
scanf("%d",&n);
printf("%d",findnthlucky(n));
29
30
31
          return 0;
32
33 }
```

