GE23131-Programming Using C-2024-2025 Week 5

ABENANTHAN P 240701005 CSE-'A'

Nested Loops – while and for , Jumps in Loops
Question 1 Simple 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

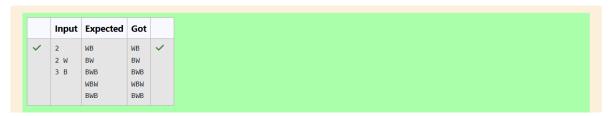
```
Input Expected Got
             WBW
                       WBW
     2
            BWB
                       BWB
      3
            WBW
                       WBW
            WBWBW
                       WBWBW
            BWBWB
                       BWBWB
             WBWBW
                       WBWBW
             BWBWB
                       BWBWB
            WBWBW
                       WBWBW
Passed all tests! <
```

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
Sample Input / Output
Input:
2
2 W
3 B
Output:
WB
BW
BWB
WBW
BWB

```
Answer: (penalty regime: 0 %)
    1 #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++)</pre>
                  scanf("%d %c",&d,&s);
for(i1=0;i1<d;i1++)
    9
   10
   11 •
                 z=(s=='W') ? 0:1;
o=(i1%2==z) ? 0:1;
   12
   13
   14
                    for(i2=0;i2<d;i2++)
   15 🔻
                      c=(i2%2==o) ? 'W' : 'B';
printf("%c",c);
   16
   17
   18
   19
                  printf("\n");
   20
   21
   22
   23 }
```



Pattern Printing

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

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 %)
   1 #include<stdio.h>
   2 v int main(){
           int n,v,p3,c,in,i,i1,i2,t,ti;
           scanf("%d",&t);
           for(ti=0;ti<t;ti++){</pre>
               scanf("%d",&n);
               printf("Case #%d\n",ti+1);
   8
               for(i=0;i<n;i++){
   9 •
  10
                  c=0;
                   if(i>0){
  11 v
                        for(i1=0;i1<i;i1++) printf("**");</pre>
  12
  13
               for(i1=i;i1<n;i1++){</pre>
  14 v
                   if(i>0) c++;
printf("%d0",++v);
  15
  16
  17
  18 (
                  p3=v+(v*(v-1))+1;
  19
  20
                    in=p3;
  21
               in=in-c;
  22
                p3=in;
  23
  24 v
                for(i2=i;i2<n;i2++){
               printf("%d",p3++);
if(i2!=n-1) printf("0");
printf("\n");
}
  25
  26
  27
  28
  29
  30 }
```

Input	Expected	Got	
✓ 3	Case #1	Case #1	~
3	10203010011012	10203010011012	
4	**4050809	**4050809	
5	****607	****607	
	Case #2	Case #2	
	1020304017018019020	1020304017018019020	
	**50607014015016	**50607014015016	
	****809012013	****809012013	
	*****10011	*****10011	
	Case #3	Case #3	
	102030405026027028029030	102030405026027028029030	
	**6070809022023024025	**6070809022023024025	
	****10011012019020021	****10011012019020021	
	*****13014017018	*****13014017018	
	******15016	******15016	
Passed all tes	ts! 🗸		

Armstrong Number

Problem Statement

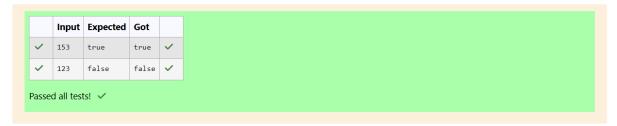
The k-digit number N is an Armstrong number if and only if the k-th power of each digit sums to N.

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.
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 %)
      #include<stdio.h>
       #include<math.h>
       int main()
   4 🔻 {
          int n;
scanf("%d",&n);
   6
          int x=0,n2=n;
          while(n2!=0)
  10
  11
               n2=n2/10;
  12
          int sum=0;
  13
  14
          int n3=n,n4;
  15
           while(n3!=0)
  16 🔻
              n4=n3%10;
  17
               sum=sum+pow(n4,x);
  18
  19
               n3=n3/10;
  20
  21
          if(n==sum)
  22 v
          {
  23
               printf("true");
  25
          else
  26 •
          {
  27
               printf("false");
  28
  29
  30
```



Reverse and Add Until Get a Palindrome

Problem Statement

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

Sample Output 1

55

Sample Input 2

789

Sample Output 2

66066

```
Answer: (penalty regime: 0 %)
   1 #include<stdio.h>
   4 v
          int rn,n,nt=0,i=0;
   6
          scanf("%d",&n);
          do{
              nt=n;rn=0;
              while(n!=0)
  10 •
                 rn=rn*10+n%10;
  11
  12
                 n=n/10;
  13
              n=nt+rn;
  15
              i++;
  16
          while(rn!=nt||i==1);
  17
  18
          printf("%d",rn);
  19
```

32 55	55	
		~
789 66066	66 66066	~

Question 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

```
1 #include<stdio.h>
      int main()
          int n=1,i=0,nt,co=0,e;
scanf("%d",&e);
while(i<e)</pre>
  5
6
7
              nt=n;
while(nt!=0)
{
 9
 11
                  co=0;
 12
                  if(nt%10!=3&&nt%10!=4)
 13 1
                  co=1;
break;
 14
 15
 16
17
                   nt=nt/10;
 18
 19
               if(co==0)
 20
                   i++;
 22
 23
               n++;
 24
 25
26 }
          printf("%d",--n);
      Input Expected Got
 ✓ 34 33344
                      33344 🗸
Passed all tests! 🗸
```