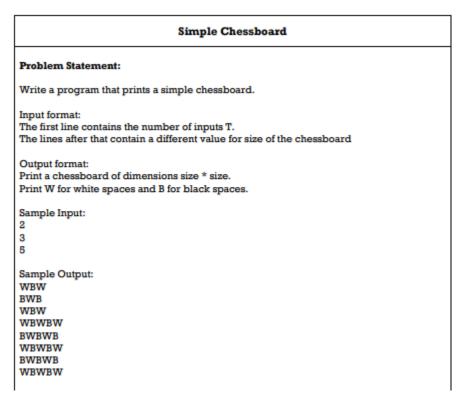
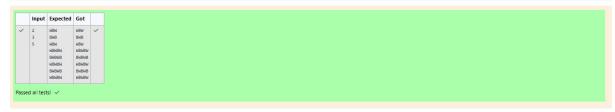
NAME: LAKSHMI.G

REGISTER NUMBER: 240801172

WEEK-05-01







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:

2

2 W 3 B

Sample Output:

WB

BW

BWB WBW

BWB

```
#include <stdio.h>
       int main()
 3
             int T,d,i,i1,i2,o,z;
5 6 7
            char c,s;
scanf("%d",&T);
for (i=0;i<T;i++)</pre>
 8 1
                   scanf("%d %c",&d,&s);
for (i1=0;i1<d;i1++)
10
11
                        z=(s=='W')?0:1;
o=(i1%2==z)?0:1;
for (i2=0;i2<d;i2++)
12
13
14
15
16
                        {
    c=(i2%2==o)?'W':'B';
    printf("%c",c);
17
18
19
                        printf("\n");
20
21
22
             return 0;
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

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

Sample Input

3

4

5

Sample Output

Case #1

10203010011012

**4050809

****607

Case #2

1020304017018019020

**50607014015016

****809012013

*****10011

Case #3

102030405026027028029030

**6070809022023024025

****10011012019020021

******13014017018 *******15016

```
#include <stdio.h>
int main()
1
2
3
4
5
6
7
8
              int n,v,p3,c,in,i,i1,i2,t,ti;
scanf("%d",&t);
for(ti=0;ti<t;ti++)</pre>
                    v=0;
scanf("%d",&n);
printf("Case #%d\n",ti+1);
for (i=0;i<n;i++);</pre>
10
11
12
13
14
                           c=0;
if(i>0)
15
                           {
16
17
18
                                   for (i1=0;i1<i;i1++) printf("**");</pre>
                            }
for (i1=i;i1<n;i1++)
19
20
21
22
                                 if (i>0) c++;
printf("%d0",++v);
                            }
if(i==0)
23
24
25
26
27
28
29
                                 p3=v+(v*(v-1))+1;
in=p3;
                           }
in=in-c;
                           p3=in;
for (i2=i;i2<n;i2++)
{
30
31
32
                                 printf("%d",p3++);
if (i2!=n-1) printf("0");
32
33
34
35
36
37
38 }
                          printf("\n");
              }
```

	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	
asse	d all test	s! 🗸		

WEEK-05-02

```
Status Finished

Started Monday, 23 December 2024, 5:33 PM

Completed Duration

Duration

Finished

Monday, 23 December 2024, 5:33 PM

Tuesday, 26 November 2024, 9:31 PM

26 days 20 hours
```

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. 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

```
#includestdio.h>
#includesmath.h>
int main()

4. {
    int n;
    6 scanf("dd",8n);
    int x=0,n2=n;
    while(n2!=0)
    9. {
    int sum = 0;
    int sum = 0;
    int sum = 0;
    int int sum =
```

```
Input Expected Got

153 true true 
123 false false 
Passed all tests! 

Passed service  

Input Expected Got

Passed service  

False  

F
```

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

32

Sample Output 1

Sample Input 2

789

Sample Output 2

66066

In	put	Expected	Got	
✓ 32	2	55	55	~
✓ 78	39	66066	66066	~
Passed all	II tests	s! ~		

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

```
| #includecstdio.h>
| 2 | int main() |
| 3 | {
| 4 | int n=1,i=0,nt,co=0,e; |
| 5 | scanf("%d",8e); |
| while(d:e) |
| 7 | {
| 8 | nt=n; |
| 9 | while(nt|=0) |
| 10 | {
| 11 | co=0; |
| 12 | if(nt%10!=3 && nt%10!=4) |
| 13 | + {
| 14 | co=1; |
| 15 | break; |
| 16 | }
| 17 | nt=nt/10; |
| 18 | }
| 19 | if(co=0) |
| 20 | + {
| 21 | i+; |
| 22 | }
| 23 | n++; |
| 24 | }
| 5 | printf("%d",-n); |
| return 0; |
| 27 | 7 |
```

