WEEK 5-1

NAME : MANISHRAJ A

ROLL NO : 240801192

1. 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

Status	Finished
Started	Monday, 23 December 2024, 5:33 PM
Completed	Saturday, 30 November 2024, 11:54 AM
Duration	23 days 5 hours

```c
#include <stdio.h>
int main()
{
    int n;
    scanf("%d",&n);
    while(n--) {
        int i;
        scanf("%d",&i);
        for (int j=1;j<=i;j++){
            if(j%2!=0) {
                char a = 'W';
                int c = 1;
                while(c<=i){
                    printf("%c",a);
                    a=(a=='W') ? 'B' : 'W';
                    c++;
                }
                printf("\n");
            }
            else {
                char z = 'B';
                int m=1;
                while(m<=i) {
                    printf("%c",z);
                    z=(z=='W')?'B':'W';
                    m++;
                }
                printf("\n");

            }
        }
    }
}

```

	Input	Expected	Got	
✓	2	WBW	WBW	✓
	3	BWB	BWB	
	5	WBW	WBW	
		WBWBW	WBWBW	
		BWBWB	BWBWB	
		WBWBW	WBWBW	
		BWBWB	BWBWB	
		WBWBW	WBWBW	

2. 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

```c
#include <stdio.h>
int main()
{
    int n;
    scanf("%d",&n);
    while(n--){
        int i;
        char ch;
        scanf("%d %c",&i,&ch);
        for(int j=1;j<=i;j++){
            if(ch=='W'){
                int c=1;
                char a='W';
                while(c<=i) {
                    printf("%c",a);
                    c++;
                    a=(a=='W')?'B':'W';
                }
                ch=(ch=='W')?'B' : 'W';
                printf("\n");

            }
            else {
                int c=1;
                char a = 'B';
                while(c<=i) {
                    printf("%c",a);
                    c++;
                    a=(a=='W')?'B' : 'W';
                }
                ch=(ch=='w')?'B':'W';
                printf("\n");
            }
        }
    }
}
```

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

3. 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: 10203040170180190 20 **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 49 5
   Output
   Case #1
   10203010011012
   **4050809
   ****607
   Case #2
   10203040170180190 20
   **50607014015016
   ****809012013
   ******10011
   Case #
   10203040502602702802903 0
   **60708090220230240 25
   ****100110120190200 21
   ******13014017018
   ********15016

```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 10203040502602702802 9030 **6070809022023024025 ****10011012019020021 ******13014017018 ********15016	Case #1 10203010011012 **4050809 ****607 Case #2 1020304017018019020 **50607014015016 ****809012013 ******10011 Case #3 10203040502602702802 9030 **6070809022023024025 ****10011012019020021 ******13014017018 ********15016	✓

Passed all tests! ✓

WEEK 5-2

1. 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 52 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

Status	Finished
Started	Monday, 23 December 2024, 5:33 PM
Completed	Friday, 6 December 2024, 9:28 PM
Duration	16 days 20 hours

```c
#include <stdio.h>
#include <math.h>
int main()
{
    int n,d,count=0,arm=0;
    scanf("%d",&n);
    d=n;
    while(d!=0){
        count++;
        d/=10;
    }
    d=n;
    for(int i=1;i<=count;i++) {
        int c=d%10;
        arm=arm+pow(c,count);
        d/=10;
    }
    if(arm==n) {
        printf("true");
    }
    else {
        printf("false");
    }
}
```

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

Passed all tests! ✓

2. 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.

```c
#include <stdio.h>
int rev(int);
int pal(int);
int main()
{
    int n,c=0;
    scanf("%d",&n);
    while(c==0) {
        int f =rev(n);
        c=pal(f);
        n=f;
    }
}
int pal(int a){
    int pa=0,b;
    b=a;
    while(b!=0) {
        pa=pa*10+b%10;
        b/=10;
    }
    if(pa==a) {
        printf("%d",a);
        return 1;
    }
    else{
        return 0;
    }
}
int rev(int a) {
    int re=0,b;
    b=a;
    while(b!=0) {
        re=re*10+b%10;
        b/=10;
    }
    return a+re;
}
```

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

Passed all tests! ✓

3. 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.

```c
#include <stdio.h>
int main()
{
    long int i,j;
    int r,n,c=0,f;
    scanf("%d",&n);
    for(i=1;c<=n;i++){
        f=0;
        j=i;
        while(j>0){
            r=j%10;
            if(r==3 || r==4) {
                j=j/10;
            }
            else{
                f=1;
                break;
            }
        }
        if(f==0) {
            c++;
            if(c==n) {
                break;
            }
        }
    }
    printf("%ld",i);
}
```

	Input	Expected	Got	
✓	34	33344	33344	✓

Passed all tests! ✓