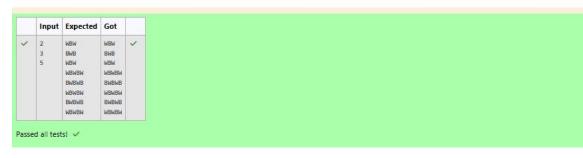
NAME : B.RAKESH
REGISTER NUMBER : 241501162
WEEK 5
Q) Write a program that prints a simple chessboard.
Input format:
The first line contains the number of inputs T.
The lines after that contain a different values for size of the chessboard
Output format:
Print a chessboard of dimensions size * size. 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



Q) 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

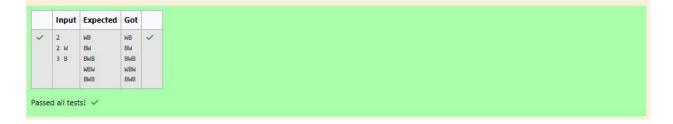
Output: WB BW

WBW

BWB

BWB

```
9
10
11
12
13
14
15
16
17
18
19
20
21
      return 0;
22 }
```



Q) 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

```
#include <stdio.h>
int main(){
              int n,v,p,c,in,i,j,k,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>
                            c=0;
if(i>0){
10
11
                                  for(j=0;j<i;j++){
    printf("**");
}</pre>
12
13
14
15
                            for(j=i;j<n;j++){
    if(i>0) c++;
    printf("%d0",++v);
16
17
18
19
                            if(i==0){
 p=v+(v*(v-1))+1;
 in=p;
20
21
22
23 24
                            in=in-c;
                          in=in-c,
p=in;
for(k=i;k<n;k++){
    printf("%d",p++);
    if(k!=n-1) printf("0");</pre>
25
26
27
28
29
30
31
                            } printf("\n");
32
33
              return 0:
```

```
Input Expected
                                         Got
              Case #1
                                         Case #1
              10203010011012
                                        10203010011012
              **4050809
                                         **4050809
              ****607
                                         ****607
              Case #2
                                        Case #2
              1020304017018019020
                                        1020304017018019020
              **50607014015016
                                         **50607014015016
              ****809012013
                                         ****809012013
              *****10011
                                         *****10011
              Case #3
                                        Case #3
              102030405026027028029030 102030405026027028029030
              **6070809022023024025
                                         **6070809022023024025
             ******13014017018
*******15016
                                        *****13014017018
******15016
Passed all tests! 🗸
```

\mathbf{Q}) 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

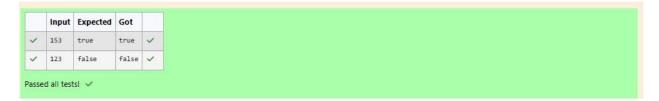
Note:

1 <= N <= 10^8

```
#include <stdio.h>
#include (math.h>
int main(){
    int n;
    scanf("%o",&n);
    int x=0, n2=n;
    while(n2!=0){
        x++;
        n2=n2/10;
    int n3=n,n4;
    while(n3!=0){
        int n3=n,n4;
    while(n3!=0){
        sum=ush+pow(n4,x);
        n3=n3/10;
    }

    if(n==sum){
        printf("true");
    } else(
        printf("false");
    }

    return 0;
}
```



Sample Input 2 789 Sample Output 2 66066

```
#include <stdio.h>
 2 v int main(){
        int rn,n,nt=0,i=0;
scanf("%d",&n);
3
 4
 5 +
        do{
 6
            nt=n;rn=0;
            while(n!=0){
                rn=(rn*10)+(n%10);
 8
 9
                n=n/10;
10
            n=nt+rn;
11
12
            i++;
13
        while(rn!=nt || i==1);
14
        printf("%d",rn);
15
16
        return 0;
17 }
```

	Input	Expected	Got	
~	32	55	55	~
~	789	66066	66066	~

Q) 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>
  2 * int main(){
         int n=1,i=0,nt,co=0,e;
scanf("%d",%e);
  3
  4
          while(i<e){
  6
              nt=n;
              while(nt!=0){
  7 +
  8
                  co=0;
                  if(nt%10!=3 && nt%10!=4){
  9 +
 10
                      co=1;
 11
                      break;
 12
                  nt=nt/10;
 13
 14
              if(co==0){
 15 +
 16
                  i++;
 17
 18
              n++;
 19
          printf("%d",--n);
 20
 21
          return 0;
```

