RAJALAKSHMI ENGINEERING COLLEGE

An Autonomous Institution, Affiliated to Anna University Rajalakshmi Nagar, Thandalam - 602 105

Programming Using C

WEEK 05

2024-2025

By

M.KESAVAN B.Tech-AIML 241501083

Status Finished Started Monday, 23 December 2024, 5:33 PM Completed Friday, 20 December 2024, 9:27 AM **Duration** 3 days 8 hours Question 1 Write a program that prints a simple chessboard. Correct Marked out of 3.00 Input format: ₹ Flag question 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

```
Answer: (penalty regime: 0 %)
     #include <stdio.h>
int main(){
   int t,a;
   scanf("%d",&t);
   for(int i=0;i<t;i++)
}</pre>
    2 * 3
    4
     6
                      scanf("%d",&a);
for(int j=0;j<a;j++)</pre>
     8
     9
   10
                             for(int k=0; k<a; k++)
   11 v
12
                                   if((j+k)\%2==0)
   13 v
14
15
                                         printf("W");
                                   }
else
{
   16
   17
   18
   19
                                               printf("B");
   20
21
22
23
24
25
                                         }
                                   printf("\n");
   26
27
                      return 0;
   28
```

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

Question **2**Correct
Marked out of 5.00

 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>
              main (,
int t,n;
char a,b;
scanf("%d",&t);
for(int i=0;i<t;i++){
    scanf("%d %c",&n,&a);
    for(int j=0;j<n;j++){
        for(int k=0;k<n;k++){
            if((j+k)%2==0){
                b=(a=='W')?'B':'W';
                printf("%C",a);}
            else{
                printf("%c",b)</pre>
             int main ()
  3 ,
   5
   6
   8
   9
 10
 11
 12
 13
 14
 15
16
 17
 18
 19
                                      }
20
21
22
23
                         return 0;
            }
 24
```

```
Input Expected Got

V 2 WB WB BW BW BW BWB WBW BWB BWB BWB
```

Question 3 Correct	Decode the logic and print the Pattern that corresponds to given input.
Marked out of 7.00	If N= 3
Flag question	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 3 *
           int main()
                  int n,v,p3,c,in,i,i1,i2,t,ti;
scanf("%d",&t);
for(ti=0;ti<t;ti++){</pre>
     4
     5
                          v=0;
                         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("**");
    }
}
    10
    11
    12
    13
    14
                                         for(i1=i;i1<n;i1++){
    if(i>0)c++;
    printf("%d0",++v);
    15 🔻
    16
    17
    18
    19
                                         if(i==0){
                                                p3=v+(v*(v-1))+1;
    20
    21
                                                in=p3;
    22
                                        in=in-c;
    23
                                       p3=in;
for(i2=i;i2<n;i2++){
    printf("%d",p3++);
    if(i2!=n-1) printf("0");</pre>
    24
25
    26
27
   28
29
30
31
32
33
                                        }printf("\n");
                         }
                  }
    34
```

	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	

Status Finished Started Monday, 21 December 2024, 5:33 PM Completed Started Monday, 22 December 2024, 7:25 PM Duration 1 Ouestion 1 Control Marked during a positive integer N, return true if and only if it is an Armstrong number. Example 1: Input: Input: Itrue Explanation: 153 is a 3-digit number, and 153 = 1^3 + 5^3 + 3^3. Example 2: Input: Input: 123 Output: false Example 3: Input:	Started Monday, 23 December 2024, 7:25 PM Duration 1 day 22 hours Correct John Tiday 22 hours The k-digit number N is an Armstrong number if and only if the k-th power of each digit sums to N. Marked out at 2,000 pm and 2,0						
Completed Take A 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: Input: Input: Itrue Explanation: It3 is a 3-digit number, and 153 = 1^3 + 5^3 + 3^3. Example 2: Input: Input: It3 Output: It3 Output: It3 Example 2: Input: It3 Output: It3 Output: It3 It3 It3 It3 It3 It3 It3 I	Outside 1 day 22 hours 1 hours						
Outside 1 day 22 hours Question 1 Correct Analysis and any agreements of a continuation of the children of the children of the children of a continuation of a continuation of a continuation of the children of the children of the children of a continuation of the children of the child	Question 1 Currect Currect Currect Shap Question The k-digit number N is an Armstrong number if and only if the k-th power of each digit sums to N. Minded out of Shap Question Shap Example 1:						
The k-digit number N is an Armstrong number if and only if the k-th power of each digit sums to N. The k-digit number N is an Armstrong number if and only if it is an Armstrong number. Example 1: Input: Input: Input: Itrue Explanation: It3 is a 3-digit number, and It3 = 1^3 + 5^3 + 3^3. Example 2: Input: I	The k-digit number N is an Armstrong number if and only if the k-th power of each digit sums to N. Marked and et al. 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:						
The k-digit number N is an Armstrong number if and only if the k-th power of each digit sums to N. Marked out of 3.00 Given a positive integer N, return true if and only if it is an Armstrong number. Example 1: Input: Input: Itrue 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:	The k-digit number N is an Armstrong number if and only if the k-th power of each digit sums to N. Maked out ed 200 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: 1534 Output:	Durai	Tation 1 day 22 nours				
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: 123 Output: 123 Example 3: Input:	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:	Correct	The k-digit number N is an Armstrong number if and only if the k-th power of each digit sums to N.				
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:	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:	3.00 ♥ Flag	Given a positive integer N, return true if and only if it is an Armstrong number.				
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:	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:	question	Example 1:				
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:	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 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:	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:						
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:	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:						
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:	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:		nuc.				
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:	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:		Explanation:				
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:	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:		153 is a 3-digit number, and 153 = 1^3 + 5^3 + 3^3.				
Input: 123 Output: false Explanation: 123 is a 3-digit number, and 123 != 1^3 + 2^3 + 3^3 = 36. Example 3: Input:	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:						
123 Output: false Explanation: 123 is a 3-digit number, and 123!=1^3 + 2^3 + 3^3 = 36. Example 3: 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:		Example 2.				
Output: false Explanation: 123 is a 3-digit number, and 123 != 1^3 + 2^3 + 3^3 = 36. Example 3: Input:	Output: false Explanation: 123 is a 3-digit number, and 123 != 1^3 + 2^3 + 3^3 = 36. Example 3: Input: 1634 Output:		Input:				
false Explanation: 123 is a 3-digit number, and 123 != 1^3 + 2^3 + 3^3 = 36. Example 3: Input:	false Explanation: 123 is a 3-digit number, and 123 != 1^3 + 2^3 + 3^3 = 36. Example 3: Input: 1634 Output:		123				
Explanation: 123 is a 3-digit number, and 123 != 1^3 + 2^3 + 3^3 = 36. Example 3: Input:	Explanation: 123 is a 3-digit number, and 123 != 1^3 + 2^3 + 3^3 = 36. Example 3: Input: 1634 Output:						
123 is a 3-digit number, and 123 != 1^3 + 2^3 + 3^3 = 36. Example 3: Input:	123 is a 3-digit number, and 123 != 1^3 + 2^3 + 3^3 = 36. Example 3: Input: 1634 Output:						
Example 3: Input:	Example 3: Input: 1634 Output:	Explanation:					
Input:	Input: 1634 Output:		123 is a 3-digit number, and 123 != 1^3 + 2^3 + 3^3 = 36.				
	1634 Output:		Example 3:				
1624	Output:		Input:				
1034			1634				
Output:	true		Output:				
true			true				

Note:

1 <= N <= 10^8

Answer: (penalty regime: 0 %)

```
#include <stdio.h>
#include <math.h>
int main()

4 v {
    int n:
           int n;
scanf("%d",&n);
int x=0,n2=n;
while(n2!=0)
 5
 6
7
8
 9 ,
                 x++;
n2/=10;
10
11
12
            int sum=0,n3=n,n4;
13
            while(n3!=0)
{
14
15
                 n4=n3%10;
sum=sum+pow(n4,x);
16
17
18
                  n3/=10;
           }
if(n==sum)
19
20
21
22
23
24
25
26
27
                printf("true");
            else
            {
                 printf("false");
28 29 }
            return 0;
```

	Input	Expected	Got	
~	153	true	true	~
~	123	false	false	~

Question **2**Correct
Marked out of 5.00

Flag
question

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

Answer: (penalty regime: 0 %)

```
1 |#include <stdio.h>
    int main()
{
 2
 4
        int rn,n,nt=0,i=0;
scanf("%d",&n);
 5
 6
        {
 8
           nt=n;
            rn=0;
while(n!=0)
 9
10
            {
11 🔻
12
                 rn=rn*10 + n%10;
13
14
15
                 n/=10;
             n=nt+rn;
16
             i++;
17
18
        while(rn!=nt ||i==1);
19 🔻
             printf("%d",rn);
20
21
22
         return 0;
23 }
```

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

Question **3**Correct
Marked out of 7.00
F Flag

question

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

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
    int main()
 3
        int n=1,i=0,nt,co=0,e;
scanf("%d",&e);
 6
        while(i<e)
 8
             nt=n;
             while(nt!=0)
10
                co=0;
11
                if(nt%10!=3 && nt%10!=4)
12
13
14
                     co=1:
15
                     break;
16
17
             nt=nt/10;
18
19
            if(co==0)
20
21
              i++;
22
23
            n++;
24
25
         printf("%d",--n);
26
          return 0;
27 }
```