

Nested Loops - while and for, Jumps in Loops

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 value for size of the chessboard

Output format:

Print a chessboard of dimensions size * size.
Print W for white spaces and B for black spaces.

Sample Input:

2
3
5

Sample Output:

WBW
BWB WBW
WBWBW
BWBWB WBWBW
BWBWB
WBWBW

Program:

```

1  #include<stdio.h>
2  int main()
3  {
4      int T,size;
5      scanf("%d",&T);
6      while(T--){
7          scanf("%d",&size);
8          for(int i=0;i<size;i++){
9              for(int j=0;j<size;j++){
10                 printf("%c",(i+j)%2==0?'W':'B');
11             }
12             printf("\n");
13         }
14     }
15     return 0;
16 }
17

```

	Input	Expected	Got	
✓	2	WBW	WBW	✓
	3	BWB	BWB	
	5	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:

2

2 W

3 B

Sample Output:

WB BW

BWB

WBW

BWB

Program:

```

1  #include<stdio.h>
2  int main()
3  {
4      int T,d,i,i1,i2,o,z;
5      char c,s;
6      scanf("%d",&T);
7      for(i=0;i<T;i++)
8      {
9          scanf("%d %c",&d,&s);
10         for(i1=0;i1<d;i1++)
11         {
12             z=(s=='W') ? 0:1;
13             o=(i1%2==z) ? 0:1;
14             for(i2=0;i2<d;i2++)
15             {
16                 c=(i2%2==o) ? 'W' : 'B';
17                 printf("%c",c);
18             }
19             printf("\n");
20         }
21     }
22     return 0;
23 }
24

```

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

Passed all tests! ✓

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: $2 \leq N \leq 100$

Input Format

First line contains T, the number of test cases, each test case contains a single integer N

Output Format

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

Sample Input

```
3
3
4
5
```

Sample Output

```
Case #1
10203010011012
**4050809
****607
Case #2
1020304017018019020
**50607014015016
****809012013
*****10011
Case #3
102030405026027028029030
**6070809022023024025
****10011012019020021
*****13014017018
*****15016
```

Program:

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

```

29     }
30     }return 0;
31 }
32
33

```

	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	

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: $1 \leq N \leq 10^8$

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

Program:

Answer: (penalty regime: 0 %)

```
1  #include<math.h>
2  #include<stdio.h>
3  int main()
4  {
5      int n;
6      scanf("%d",&n);
7      int x=0,n2=n;
8      while(n2!=0)
9      {
10         x++;
11         n2=n2/10;
12     }
13     int sum=0;
14     int n3=n,n4;
15     while(n3!=0)
16     {
17         n4=n3%10;
18         sum=sum+pow(n4,x);
19         n3=n3/10;
20     }
21
22
23     if(n==sum)
```

```
22
23     if(n==sum)
24     {
25         printf("true");
26     }
27     else{
28         printf("false");
29     }
30     return 0;
31 }
```

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

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

$1 \leq \text{num} \leq 99999999$

Sample Input 1

32

Sample Output 1

55

Sample Input 2

789

Sample Output 2

66066

Program:

```

1  #include<stdio.h>
2  int main()
3  {
4      int rn,n,nt=0,i=0;
5      scanf("%d",&n);
6      do
7      {
8          nt=n;rn=0;
9          while(n!=0)
10         {
11             rn=rn*10+n%10;
12             n=n/10;
13         }
14         n=nt+rn;
15         i++;
16     }
17     while(rn!=nt || i==1);
18     printf("%d",rn);
19     return 0;
20 }
21

```

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

Passed all tests! ✓

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

Program:

```

1  #include<stdio.h>
2  int main()
3  {
4      int n=1,i=0,nt,co=0,e;
5      scanf("%d",&e);
6      while(i<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     }
25     printf("%d",--n);
26     return 0;
27 }

```

	Input	Expected	Got	
✓	34	33344	33344	✓

Passed all tests! ✓