**Free Placement Preparation Online Course with Free Mock Test
visit:** www.MyPlacementPrep.com

www.MyPlacementPrep.com

Pro Material Series

Join Telegram Channel for more updates:
https://t.me/MyPlacementprepApp

Visit www.MyPlacementPrep.com .
India's No1 website for Placement Materials and Free mock test Series
Free Placement Preparation Online Course video now available.

Join Telegram Channel: https://t.me/MyPlacementprepApp
Join Telegram Group: https://t.me/myPlacementPrep

Coding Question -1 :

Please comment down the code in other languages as well below –

C Program to check if two given matrices are identical

```cpp
*code language="cpp"+ #include <stdio.h> #define N 4

// This function returns 1 if A[][] and B[][] are identical // otherwise returns 0
int areSame(int A[][N], int B[][N])
{

int i, j;
for (i = 0; i < N; i++)

for (j = 0; j < N; j++) if (A[i][j] != B[i][j])

return 0; return 1;

}

int main() {

int A[N][N] = { {1, 1, 1, 1}, {2, 2, 2, 2},

{3, 3, 3, 3}, {4, 4, 4, 4}};

int B[N][N] = { {1, 1, 1, 1}, {2, 2, 2, 2},

{3, 3, 3, 3}, {4, 4, 4, 4}};

if (areSame(A, B)) printf("Matrices are identical");

else
printf("Matrices are not identical");

return 0; }
```

June 24, 2019

Print a given matrix in spiral form

Given a 2D array, print it in spiral form. See the following examples. Please comment down the code in other languages as well below –

Input: 1234

5678
9 101112 13 14 15 16

Output:
1 2 3 4 8 12 16 15 14 13 9 5 6 7 11 10 Input:

123456
7 8 9101112 13 14 15 16 17 18

Output:
1 2 3 4 5 6 12 18 17 16 15 14 13 7 8 9 10 11

Code in C
*code language="cpp"+
#include
#define R 3
#define C 6
void spiralPrint(int m, int n, int a[R][C]) {
int i, k = 0, l = 0;
/* k – starting row index
m – ending row index
l – starting column index
n – ending column index

```
i – iterator
*/
while (k < m && l < n) {
/* Print the first row from the remaining rows */
for (i = l; i < n; ++i)
{
printf("%d ", a[k][i]);
}
k++;
/* Print the last column from the remaining columns */ for (i = k; i < m;
++i)
{
printf("%d ", a[i][n-1]);
}
n–;
/* Print the last row from the remaining rows */
if ( k < m)
{
for (i = n-1; i >= l; –i)
{
printf("%d ", a[m-1][i]);
}
m–;
}
/* Print the first column from the remaining columns */ if (l < n)
{
for (i = m-1; i >= k; –i)
{
printf("%d ", a[i][l]);
}
l++;
}
}
}
```

```
/* Driver program to test above functions */ int main()
{
int a[R][C] = { {1, 2, 3, 4, 5, 6},

{7, 8, 9, 10, 11, 12}, {13, 14, 15, 16, 17, 18} };
spiralPrint(R, C, a); return 0;
}
[/code]
```

1 2 3 4 5 6 12 18 17 16 15 14 13 7 8 9 10 11

```
Code in Java
*code language="java"+
private static int[] spiral(int[][] matrix)
{ //Test if matrix is rectangular
int len=matrix[0].length;
for(int i=1; i&lt;matrix.length; i++)
{
if(matrix[i].length!=len)
{
System.out.println("Not rectangular"); return null; }
}
int[] erg = new int[len*matrix.length];
int[] borders = new int[]{0,0,matrix.length-1, len-1}; int[] pointer = new
int[]{0,0};
int state=0;
for(int i=0; i&lt;erg.length; i++)
{
erg[i]=matrix[pointer[0]][pointer[1]];
switch (state)
{
case 0:
if(pointer[1] == borders[3])

{
state++;
```

```
pointer[0]++;
borders[0]++;
break;
}
pointer[1]++;
break;
case 1:
if(pointer[0] == borders[2])
{ state++; pointer[1]–; borders[3]–; break;
}
pointer[0]++;
break;
case 2: if(pointer[1] == borders[1]) {
state++; pointer[0]–;
borders[2]–; break; } p
ointer[1]–; break;
case 3:
if(pointer[0] == borders[0]){ state=0; pointer[1]++; borders[1]++;
break;
}
pointer[0]–;
break;
}
}
return erg;
}
[/code]
```

June 24, 2019

Coding Question – 3

0's.

Please comment down the code in other languages as well below –

```cpp
*code language="cpp"+ #include <stdio.h> #include <stdlib.h>
#include<conio.h> #define COL 4

#define ROW 4
using namespace std; int main()
{
int arr[ROW][COL]= { {1,1,1,1},
{1,1,0,0},
{1,0,0,0},
{1,1,0,0},
};

int rownum;
int i = 0, j = COL-1; while(i<ROW && j>;0) {
if(arr[i][j]==0)
{
j—;
rownum=i;}
else
i++;
} printf("%d",rownum); getch();
return 0;
```

Given an n-by-n matrix of 0's and 1's where all 1's in each row come before all

0's, find the most efficient way to return the row with the maximum number of

```
} [/code]
```

June 24, 2019
AMCAT Coding Question 4 (Unsolved)

limit.

[222] A Pythagorean triplet is a set of three integers a, b and c such that a + b = c . Given a limit, generate all Pythagorean Triples with values smaller than given

Input : limit = 20 Output : 3 4 5
8 6 10
5 12 13 15 8 17 12 16 20

A Simple Solution is to generate these triplets smaller than given limit using three nested loop. For every triplet, check if Pythagorean condition is true, if true, then print the triplet. Time complexity of this solution is $O(limit^3)$ where 'limit' is given limit.

An Efficient Solution can print all triplets in O(k) time where k is number of triplets printed. The idea is to use square sum relation of Pythagorean triplet, i.e., addition of squares of a and b is equal to square of c, we can write these number in terms of m and n such that,

[22] a= m -n

b=2*m*n

$c = m^2 + n^2$ because,
$a^2 = m^4 + n^4 - 2 * m^2 * n^2$ $b^2 = 4*m^2*n^2$
$c^2 = m^4 + n^4 + 2*m^2*n^2$

We can see that $a^2 + b^2 = c^2$, so instead of iterating for a, b and c we can iterate for m and n and can generate these triplets.

Below is C implementation of above idea.

*code language="cpp"+
```
// A C program to generate pythagorean triplets // smaller than a given limit
#include <stdio.h>
#include <math.h>
```

```
// Function to generate pythagorean triplets // smaller than limit
void pythagoreanTriplets(int limit)
{

// triplet: a^2 + b^2 = c^2 int a, b, c=0;

// loop from 2 to max_limitit int m = 2;

// Limiting c would limit all a, b and c while (c < limit)
{

// now loop on j from 1 to i-1 for (int n = 1; n < m; ++n)
{

// Evaluate and print triplets using // the relation between a, b and c a = m*m
– n*n;
b = 2*m*n;

c = m*m + n*n;

if (c > limit) break;

printf("%d %d %d\n", a, b, c); }

m++; }

}

// Driver program int main()
{

int limit = 20; pythagoreanTriplets(limit); return 0;

} [/code]
```

```
345
8 6 10
5 12 13 15 8 17 12 16 20
```

Time complexity of this approach is O(k) where k is number of triplets printed for a given limit (We iterate for m and n only and every iteration prints a triplet)

June 23, 2019
1) Find the distinct elements in a given array. (Assume size of an array n<=20)
Sample Input:

- • 9 = size of an array
- • 2 3 4 5 6 1 2 3 4 = array elements

Sample Output:

- 234561

Program:

// C program to print all distinct elements in a given array

```c
#include
void distict_elements(int a[], int n); int main()
{
int size_array, i, arr[20];
// Get the array size
scanf("%d", &size_array)
// Get the array elements
for(i=0; i<size_array; i++)
{
scanf("%d", &arr*i+);
}

// Function call to print the distinct elements in an array distict_elements(arr,
size_array);
return 0;
}
```

```
void distict_elements(int a[], int n) {
int i, j;
// Pick all elements one by one

for (i=0; i<n; i++)
{
// Check if the picked element is already printed for (j=0; j<i; j++)
{
if (a[i] == a[j])
break;
}

// If not printed earlier, then print it if (i == j)
printf("%d ", a*i+);
}

}
```

2) Program to sort array in ascending & descending order.

Input:

5 86927 Output: 26789 98762

Program:

```
// C program to sort the given array elements in ascending and descending
order #include
int main(void)
{

int arr[10], i=0, j=0, size, temp;
// Get the size of an array
scanf ("%d", &size);
// Get the array elements as an input for (i = 0; i <size; i++)
```

```
{
scanf ("%d", &arr*i+);
}
// Sorting elements in ascending order
for (j=0 ; j<(size-1) ; j++)
{
for (i=0 ; i<(size-1) ; i++)
{
if (arr[i+1] < arr[i])
{
temp = arr[i];
arr[i] = arr[i + 1];
arr[i + 1] = temp;
}
}
}
// Print the elements from index value 0 to (size-1) –> ascending order for (i=0
; i {
printf ("%d ", arr*i+);

} printf("\n");

// Print the elements from the index value (size-1) to 0 –> descending order for
(i=size-1; i>=0 ; i–)
{
printf ("%d ", arr*i+);

}
return 0; }
```

3) Sort first half in ascending and second half in descending order.

Example 1:

Input:
8 24793168 Output: 12349876 Example 2: Input:

6
123456 Output: 123654

Algorithm:

i) Sort the given array.
ii) Run a loop up to half the length of the array and print the elements of the sorted array.
iii) Run a loop from the last index of the array to the middle of the array and print the elements in reverse order.

Program:

```
#include
void sorting_elements(int arr[], int n);
void display(int arr[], int n);
int main()
{
int size, arr[20], i;
scanf("%d", &size);
for(i=0; i<size; i++)
{
scanf("%d", &arr*i+);
}
display(arr, size);
return 0;
}
// Sort the elements in the ascending order void sorting_elements(int arr[], int n)
{
int i,j,temp;
for (j=0 ; j<(n-1) ; j++)
{
for (i=0 ; i<(n-1) ; i++)
{
if (arr[i+1] < arr[i])
```

```
{
temp = arr[i];
arr[i] = arr[i + 1];
arr[i + 1] = temp;
}
}
}
}
// Display the sorted elements
void display(int arr[], int n)
{
sorting_elements(arr, n);
int i, j

// Print the first half as such (i.e. from index 0 to midlle) for (i=0; i<n/2; i++)
{
printf("%d ", arr[i]);

}

// Print the second half in the reverse order (i.e. from n-1 to midlle) for (j=n-1;
j>=n/2; j–)
{
printf("%d ", arr*j+);

} }
```

<mark>4) Print the following pattern pattern</mark>

Input:

34

Output:

3
44

555
6666
555
44
3

Input :

44

Output:

4
55
666
7777
666
55
4

Program:

```c
#include
int main()
{
int i,j,s,N,count=0; scanf("%d%d",&s,&N); for(i=s;count<4;count++) {

for(j=0;j<count+1;j++) printf("%d",i); printf("\n");
i=i+1;

} for(i=s+N-2;count>0;count--) {

for(j=0;j<count-1;j++) printf("%d",i); printf("\n");
i=i-1;

}
return 0; }
```

5) Print the following pattern pattern

Input :

3

Output:

1
2*2
3*3*3
3*3*3
2*2
1

Input :

4

Output:

1
2*2
3*3*3
4*4*4*4
4*4*4*4
3*3*3
2*2
1

Program:

```
#include
int main()
{
int i,j,k,N,count=0; scanf("%d",&N); for(i=1;i<=N;i++)
```

```
{
k=1; for(j=0;j<i;j++) { printf("%d",i); if(k<i)
{
printf("*");

k=k+1;
}
}
printf("\n");
} for(i=N;i>0;i–) {

k=1; for(j=0;j<i;j++) { printf("%d",i); if(k<i)
{
printf("*"); k=k+1;
}
}
printf("\n");
}
return 0;
}
```

6) Print the below pattern

Input:

4

Output:

1
2*3
4*5*6
7*8*9*10
7*8*9*10
4*5*6
2*3
1

Program:

```
#include
int main() {
int i,j,count=1,n; printf("Enter a number\n"); scanf("%d",&n); for(i=1;i<=n;i++)
{
for(j=1;j<=i;j++)
{
if(j<i) printf("%d*",count++);
else
printf("%d",count++);
} printf("\n");
}
count=count-n; for(i=n;i>=1;i–)
{ for(j=1;j<=i;j++)
{
if(j<i) printf("%d*",count++);
else
printf("%d",count++);
}

count=(count+1)-2*i; printf("\n");
}
return 0;

}
```

7) Print the following pattern

Input:

34

Output:

3
44

555
6666
6666
555
44
3

Program:

```
#include
int main()
{
int i,j,s,N,count=0; scanf("%d%d",&s,&N); for(i=s;count<4;count++) {

for(j=0;j<count+1;j++) printf("%d",i); printf("\n");
i=i+1;

} for(i=s+N-2;count>0;count–) {
for(j=0;j<count-1;j++) printf("%d",i);
printf("\n");
i=i-1;
}
return 0;
}
```

8) Print the below pattern

Input:

5

Output:

1
3*2
4*5*6
10*9*8*7 11*12*13*14*15

Program:

```c
#include
int main()
{
int i,j,k,l=1,N,d,r,count=0; scanf("%d",&N); for(i=1;i<=N;i++)

{
k=1;
d=i%2; r=l+i-1; for(j=0;j<i;j++) {

if(d==0)
{ printf("%d",r); r--;

if(k<i)
{ printf("*"); k=k+1;

}
l++;
continue;
} printf("%d",l); l++;

if(k<i)
{ printf("*"); k=k+1;

}
} printf("\n"); }
return 0;
}
```

<mark>9) Print the below pattern</mark>

Input:

4

Output:

1*2*3*4*17*18*19*20 – -5*6*7*14*15*16
– – – -8*9*12*13
– – – – – -10*11

Program:

```
#include
void pattern(int);
int main()
{
int n;
scanf("%d", &n);
pattern(n);
return 0;
}
void pattern(int n)
{
int i, j, k, s, a = 1,b = n*n + 1; for (i = n; i >= 1; i–) {

for (s = 0; s < n – i; s++) printf("–");
for (j = 0; j < i; j++) printf("%d*", a++);

for (k = 0; k < i – 1; k++) printf("%d*", b++); printf("%d\n", b);
b -= 2*(i – 1);

} }
```

// last b should without *

10) Print pattern

Input:

3

Output:

333 313 323 333

Program:

```
#include
int main()
{
int i, j, n, c=1; scanf("%d", &n); for(i=1; i<=n+1; i++) {

for(j=1; j<=n; j++) {
if(i!=1 && j==n-1) {

printf("%d ", c); c++;
}
else

printf("%d ", n); }
printf("\n");
}

return 0; }
```

11) Paranthesis checker: Check whether the given expression is valid or not(only

parenthesis symbol).

Test Case: 1

Input: "(( ))" Output: Valid Test Case: 2 Input: "()(" Output: Invalid

Program:

```c
#include
#include
#include
int top = -1; char stack[100]; void push(char);

void pop();
void find_top();
void main()
{
int i;
char a[100];
scanf("%s", &a);
for (i = 0; a*i+ != '\0'; i++)
{
if (a*i+ == '(')
push(a[i]);
else if (a*i+ == ')')
pop();
}
find_top();
}
// to push elements in stack void push(char a)
{
top++;
stack[top] = a;
}
// to pop elements from stack

void pop()
{
if (top == -1)
{
printf("Invalid");
exit(0);
}
else
```

```
top–;
}
// to find top element of stack

void find_top()
{
if (top == -1) printf("Valid"); else printf("Invalid"); }
```

12) Print the transpose of a Matrix:

```
#include
int main()
{
int a[10][10], transpose[10][10], r, c, i, j; printf("Enter rows and columns of
matrix: "); scanf("%d %d", &r, &c);

// Storing elements of the matrix printf("\nEnter elements of matrix:\n");
for(i=0; i<r; ++i)
for(j=0; j<c; ++j)

{
printf("Enter element a%d%d: ",i+1, j+1); scanf("%d", &a*i][j]);
}
// Displaying the matrix a[][] */ printf("\nEntered Matrix: \n");
for(i=0; i<r; ++i)
for(j=0; j<c; ++j)
{
printf("%d ", a*i+*j+);
if (j == c-1)
printf("\n\n");
}
// Finding the transpose of matrix a for(i=0; i<r; ++i)
for(j=0; j<c; ++j)
{
transpose[j][i] = a[i][j];
}
```

```
// Displaying the transpose of matrix a printf("\nTranspose of Matrix:\n");
for(i=0; i<c; ++i)
for(j=0; j<r; ++j)
{
printf("%d ",transpose*i+*j+);
if(j==r-1)

printf("\n\n"); }
return 0;
}
```

13) Matrix Addition:

Program:

```
#include
int main()
{
int r, c, a[100][100], b[100][100], sum[100][100], i, j; printf("Enter number of
rows (between 1 and 100): "); scanf("%d", &r);

printf("Enter number of columns (between 1 and 100): "); scanf("%d", &c);
printf("\nEnter elements of 1st matrix:\n");
for(i=0; i<r; ++i)

for(j=0; j<c; ++j)
{
printf("Enter element a%d%d: ",i+1,j+1); scanf("%d",&a*i+*j+);
}
printf("Enter elements of 2nd matrix:\n"); for(i=0; i<r; ++i)
for(j=0; j<c; ++j)
{
printf("Enter element a%d%d: ",i+1, j+1); scanf("%d", &b*i+*j+);
}
// Adding Two matrices
for(i=0;i<r;++i)
for(j=0;j<c;++j)
```

```
{
sum[i][j]=a[i][j]+b[i][j];
}
// Displaying the result
printf("\nSum of two matrix is: \n\n"); for(i=0;i<r;++i)
{ for(j=0;j<c;++j)
{
printf("%d ",sum*i+*j+);
if(j==c-1)

{ printf("\n\n"); }
}
return 0;
}
```

July 8, 2019
Problem: To find GCD of two number. Solution:

```
#include

using namespace std; int gcd_iter(int u, int v)
{
int t;
while (v)
{
t = u;
u = v;
v = t % v;
}
return u < 0 ? -u : u;
}
int main()
{
int n=3,m=6;
int result=gcd_iter(n,m); cout<<result;
return 0;
}
```

C Program to find lcm of 3 numbers

```cpp
*code language="cpp"+ #include<stdio.h>

int lcm(int,int);
int main(){
int a,b,c,l,k;
printf("Enter any three positive integers "); scanf("%d%d%d",&a,&b,&c);

if(a<b)
l = lcm(a,b); else

l = lcm(b,a); if(l>c)
k= lcm(l,c); else

k= lcm(c,l);

printf("LCM of two integers is %d",k);

return 0;

}

int lcm(int a,int b){

int temp = a;

while(1){

if(temp % b == 0 && temp % a == 0)

break;

temp++;

}

return temp; }
```

[/code]

*code language="cpp"+

```cpp
#include<iostream> using namespace std;

int lcm(int, int, int); int hcf(int, int, int); int main()
{

int a,b,c;
int LCM, HCF;
cout<<"Enter 1st number: "; cin>>a;
cout<<"Enter 2nd number: "; cin>>b;
cout<<"Enter 3rd number: "; cin>>c;

LCM = lcm(a,b,c);
HCF = hcf(a,b,c);
cout<<"LCM of "<<a<<","<<b<<","<<c<<" is = "<<LCM<<endl; cout<<"HCF of
"<<a<<","<<b<<","<<c<<" is = "<<HCF<<endl; return 0;
}
int lcm(int x,int y, int z)
{
long max,lcom, count, flag=0;
if(x>=y&&x>=z)
max=x;
else if(y>=x&&y>=z)
max=y;
else if(z>=x&&z>=y)
max=z;
for(count=1;flag==0;count++)
{
lcom=max*count;
if(lcom%x==0 && lcom%y==0 && lcom%z==0)
{
```

```
flag=1;
}
}
return lcom; }

int hcf(int p, int q, int r)
{
int gcf=1,flag=0, count;
for(count=1; flag==0;count++)
{ if(p%count==0&&q%count==0&&r%count==0) gcf=count;

if(count>p&&count>q&&count>r) {
flag=1;
}

}
return gcf; }
[/code]
```

Java Program to Find LCM of N Numbers

LCM of three Numbers in Java

```
*code language="java"+ public class LCM {

public static long lcm(int[] numbers) { long lcm = 1;
int divisor = 2;
while (true) {

int cnt = 0;
boolean divisible = false;
for (int i = 0; i < numbers.length; i++) {
/**
* lcm (n1,n2,... 0)=0.For negative number we convert into * positive and
calculate lcm.
*/
```

```
if (numbers[i] == 0) {
return 0;
} else if (numbers[i] < 0) {
numbers[i] = numbers[i] * (-1);
}
if (numbers[i] == 1) {
cnt++;
}
/**
* divide numbers by devisor if complete division i.e. without
* remainder then replace number with quotient; used for find
* next factor
*/
if (numbers[i] % divisor == 0) {
divisible = true;
numbers[i] = numbers[i] / divisor;
}
}
/**
* If divisor able to completely divide any number from array
* multiply with lcm and store into lcm and continue to same divisor * for next
factor finding. else increment divisor
*/
if (divisible) {
lcm = lcm * divisor;
} else {
divisor++;
}
/**
* Check if all numbers is 1 indicate we found all factors and
* terminate while loop.
*/
if (cnt == numbers.length) {
return lcm;
}
```

```java
}
}

public static int lcm2(int num1, int num2) { if(num1==0 || num2==0){
return 0;
}else if(num1<0){

num1=num1*(-1); }else if(num2<0){ num2=num2*(-1); }

int m = num1;
int n = num2;
while (num1 != num2) { if (num1 < num2) { num1 = num1 + m;
} else {
num2 = num2 + n;
}
}
return num1;
}

public static void main(String[] args) {
int[] numbers = {140, 72, 130};
System.out.println("*** Least Common Multiple ***");
System.out.println("LCM(Least Common Multiple) of N numbers using Table
method ");
System.out.println(lcm(numbers));
System.out.println("LCM of two numbers using repetative addition");
System.out.println(lcm2(1, 72));

}
} [/code]
```

Output:

```
*** Least Common Multiple ***
LCM(Least Common Multiple) of N numbers using Table method 32760
LCM of two numbers using repetative addition
72
```

Ques. To print the trapezium pattern?

IfN=4 1*2*3*4*17*18*19*20

5*6*7*14*15*16 8*9*12*13

10*11 If n = 5

1*2*3*4*5*26*27*28*29*30 6*7*8*9*22*23*24*25

10*11*12*19*20*21
13*14*17*18 15*16

IfN=2 1*2*5*6

3*4

InC
```
#include
int main(){
int n=5,num=1,i=1,space=0,k=1,number=n; for(i=0;i<n;i++)
{
for(int j=1;j<=space;j++)
{

printf("-");

}
for(int m=1;m<2*n-space;m++) {
if(m%2==0) printf("%s","*");
else
printf("%d",num++);
}
printf("%s","*");
for(int l=1;l<2*n-space;l++)
{
if(l%2==0)
printf("%s","*");
```

```
else
{ printf("%d",k+number*number); k++;
}
}
number--;

space=space+2; printf("\n");
}
return 0;

}
```

In C++

```
#include
using namespace std;
int main(){
int n=4,num=1,i=1,space=0,k=1,number=n; for(i=0;i<n;i++)
{
for(int j=1;j<=space;j++)
{

cout<<"-";

}
for(int m=1;m<2*n-space;m++) {
if(m%2==0) cout<<"*"; else cout<<num++; }

cout<<"*";
for(int l=1;l<2*n-space;l++)
{
if(l%2==0)
cout<<"*";
else
{ cout<<k+number*number; k++;
}
}
number--;
```

```
space=space+2; cout<<endl;
}
return 0;

}
```

Trapezium Pattern program in Java

```java
public class Pattern {
public static void main(String[] args) {

int count1=0,count2=0;
int N=4;
for(int i=N;i>=1;i–) {
for(int j=N;j>i;j–) System.out.print(" ");

for(int k=1;k<=i;k++) System.out.print(++count1+"*");

for(int l=1;l<=i;l++) { System.out.print(++count2+i*i); if(l!=i)
System.out.print("*");
}
System.out.println(); }
}

}
```

Ques. Print the following Pattern and get the output to match test cases?

To print the pattern like for n=3

the program should print

1112 3222 3334

Program in C++ #include <iostream>

using namespace std;

int main() {

```
int n=3,c=n-1; for(int i=1;i<=n;i++) {

if(i%2==0) cout<<c++;
for(int j=1;j<=n;j++) {

cout<<i; }

if(i%2!=0) cout<<c++; cout<<"\n";

}

return 0; }
```

Program in C

```
#include
int main(void) {
```

```
int i,j,n=3,c=n-1; for(i=1;i<=n;i++) {
if(i%2==0) printf("%d",c++); for(j=1;j<=n;j++) {

printf("%d",i); }
if(i%2!=0)
printf("%d",c++); printf("\n");
}

return 0; }
```

Code in Java

```
public class Practice{
public static void main(String[] args){ PrintPat(3); }
public static void PrintPat(int a)
{ int n=1,i,j=1;
while(n<=a){
if(n%2!=0){
```

```
for(i=1;i<=a;i++)
System.out.print(n);

System.out.print(++j); System.out.println(); }
else{ System.out.print(++j); for(i=1;i<=a;i++) System.out.print(n);
System.out.println(); }
```

n++; }}}

June 24, 2019

Ques. Programming Pattern to Print 2*N Number of rows for input Pattern?

```
3
44
555
6666
555
44 3
```

Code in C++

C/C++ Program to Print- 3 44 555 6666 6666 555 44 3

*code language="cpp"+ #include <iostream> using namespace std; int main()

```
{
int n=4,num=n-1;
for(int i=1;i<=n;i++)
{
for(int j=1;j<=i;j++)
cout<<num;
num++;
cout<<endl; } num–; for(int i=n;i>=1;i–) {
for(int j=1;j<=i;j++)
```

```
cout<<num;
num–;
cout<<endl;
}
return 0;

} [/code]
```

Please do comment the code in other languages :).

Code in Java –

Java Program to Print- 3 44 555 6666 6666 555 44 3

```
*code language="java"+ public class Pattern {

public static void main(String[] args) {

int N=4;
for(int i=1;i<=N;i++) {
for(int j=1;j<=i;j++) System.out.print(i+2); System.out.println(); } for(int i=N-1;i>=1;i–) {
for(int k=1;k<=i;k++) System.out.print(i+2);

System.out.println(); }
}

} [/code]
```

www.MyPlacementPrep.com

Free Mock Test and Video Tutorial