**Task No. 1:**

Write a code which prints the following series:

1 2 4 8 - - - - n

**Solution:**

class Program

{

static void printSeq(int counter)

{

if (counter <= 0) return; // first the exit condition

int value = counter - 1;

printSeq(value); // go down to the bottom, in order to print values from lovest values to higher

Console.Write(" " + (int)Math.Pow(2, value)); // print the value

}

static void Main()

{

Console.Write("Enter the value of n in integer: ");

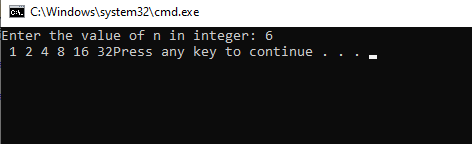
int n = Convert.ToInt32(Console.ReadLine());

printSeq(n);

}

}

**Output:**



**Task No. 2:**

Write a program which calculates the square of a number using odd number series implemented with the help of recursion concept

**Solution:**

class Program

{

static void squareSeries(int y, int x = 1)

{

if (x > y)

return;

Console.Write(x\*x + " ");

squareSeries(y, x + 2);

return;

}

static void Main(string[] args)

{

Console.WriteLine("Enter limit of series: ");

int x = int.Parse(Console.ReadLine());

Console.WriteLine("Square of odd number series from 1 to {0} is:" , x);

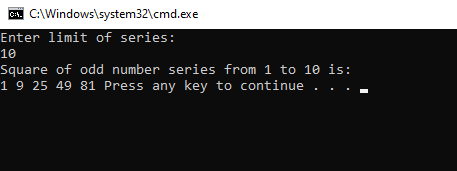
squareSeries(x);

}

}

}

**Output:**



**Task No. 3:**

Write a program which takes input of an integer number and returns the sum of all numbers. i.e., if input is 3453 then the output should be 15 (3+4+5+3).

**Solution:**

class Program

{

static int sumDigits(int no)

{

return no == 0 ? 0 : no % 10 +

sumDigits(no / 10);

}

public static void Main()

{

Console.WriteLine("Enter any integer :");

int n = Convert.ToInt32(Console.ReadLine());

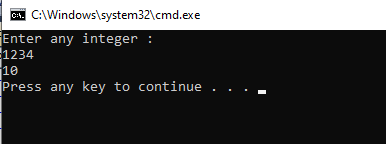
Console.WriteLine(sumDigits(n));

}

}

}

**Output:**



**Task No. 4:**

Write a program to calculate binomial coefficients of any given number using recursion.

**Solution:**

class Program

{

static int binomialCoeff(int n, int k)

{

// Base Cases

if (k == 0 || k == n)

{

return 1;

}

else

{

return binomialCoeff(n - 1, k - 1) + binomialCoeff(n - 1, k);

}

}

public static void Main()

{

int n = 8, k = 3;

Console.WriteLine("Value of C(" + n + ","

+ k + ") is " +

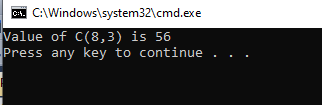
binomialCoeff(n, k));

}

}

}

**Output:**



**Task No. 5:**

Calculation of number of moves for N number of disk in Tower of Hanoi problem using recursion.

**Solution:**

static void towerOfHanoi(int n, char from\_rod,

char to\_rod, char aux\_rod)

{

if (n == 1)

{

Console.WriteLine("Move disk 1 from rod " + from\_rod

+ " to rod " + to\_rod);

return;

}

else

{

towerOfHanoi(n - 1, from\_rod, aux\_rod, to\_rod);

Console.WriteLine("Move disk " + n + " from rod "

+ from\_rod + " to rod " + to\_rod);

towerOfHanoi(n - 1, aux\_rod, to\_rod, from\_rod);

}

}

public static void Main()

{

// Number of disks

Console.WriteLine("Kindly Enter total number of disks:");

int n = int.Parse(Console.ReadLine());

// A, B and C are names of rods

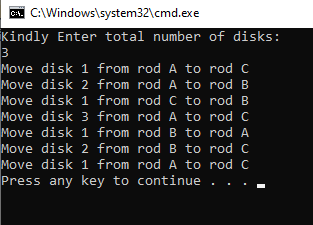
towerOfHanoi(n, 'A', 'C', 'B');

}

}

}

**Output:**



**Task No. 6:**

Write a program to calculate H.C.F of two numbers, using recursion.

**Solution:**

class Program

{

static int hcf(int a, int b)

{

// Everything divides 0

if (a == 0)

return b;

if (b == 0)

return a;

// base case

if (a == b)

return a;

// a is greater

if (a > b)

{

return hcf(a - b, b);

}

else

{

return hcf(a, b - a);

}

}

public static void Main()

{

Console.WriteLine("Kindly enter first number:");

int a = int.Parse(Console.ReadLine());

Console.WriteLine("Kindly enter second number:");

int b = int.Parse(Console.ReadLine());

Console.WriteLine("HCF of "+ a + " and " + b + " is " + hcf(a, b));

}

}

}

**Output:**

