**Task No. 1:** Write a code which prints the following series:

2 4 8 - - - - n

**Solution:**

public static void series(int start, int end)

{

if (end == 0)

{

return;

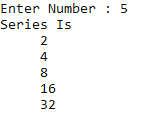
}

end--;

**Output:**

Console.WriteLine(" " + 2 \* start);

series(2 \* start, end);

****

}

static void Main(string[] args)

{

Console.Write("Enter Number : ");

int num1 = int.Parse(Console.ReadLine());

Console.WriteLine("Series Is ");

series(1, num1);

}

**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:**

public static void Oddseries(int start, int end)

{

if (start>end)

{

return;

}

Console.WriteLine($"\t{start}\t\t{Math.Pow(start, 2)}\n ");

start += 2;

Oddseries(start, end);

}

static void Main(string[] args)

{

Console.Write("Enter Number : ");

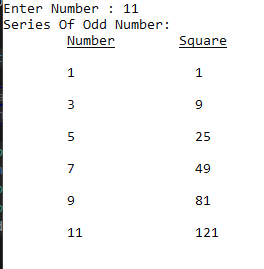
int num1 = int.Parse(Console.ReadLine());

Console.WriteLine("Series Of Odd Number: ");

Console.WriteLine($"{UNDERLINE}\tNumber\t{RESET} {UNDERLINE}Square{RESET}\n");

Oddseries(1, num1);

}

**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:**

public static int Sumnumber(int number)

{

if (number==0)

{

return 0;

}

return (number % 10 + Sumnumber(number / 10));

}

static void Main(string[] args)

{

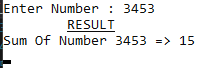
Console.Write("Enter Number : ");

int num1 = int.Parse(Console.ReadLine());

Console.WriteLine($"{UNDERLINE}\tRESULT\t{RESET}");

Console.WriteLine($"Sum Of Number {num1} => " +Sumnumber(num1));

}

**Output:**

**Task No. 4:** Calculation of number of moves for N number of disk in Tower of Hanoi problem using recursion.

**Solution:**

public static void Towerofhonei(int n, string src, string helper, string destination,ref int count)

{

if (n==1)

{

Console.WriteLine($"Moving {n} From { src} To {destination} ");

count++;

return;

}

Towerofhonei(n-1,src,destination,helper,ref count);

Console.WriteLine($"Moving {n} From { src} To {destination} ");

count++;

Towerofhonei(n-1,helper,src,destination,ref count);

}

static void Main(string[] args)

{

int n;int count = 0;

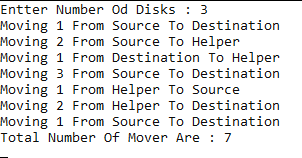
Console.Write("Entter Number Od Disks : ");

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

Towerofhonei(n,"Source","Helper","Destination",ref count);

Console.WriteLine( "Total Number Of Mover Are : {0}",count);

}

**Output:**

**Task No. 5:** Write a program to calculate H.C.F of two numbers, using recursion.

**Solution:**

static long gcd(long n1, long n2)

{

if (n2 == 0)

{

return n1;

}

else

{

return gcd(n2, n1 % n2);

}

}

static void Main(string[] args)

{

long num1, num2, hcf, lcm;

Console.WriteLine("\n\n Recursion : Find the LCM and GCD of two numbers :");

Console.WriteLine("------------------------------------------------------");

Console.Write(" Input the first number : ");

num1 = Convert.ToInt64(Console.ReadLine());

Console.Write(" Input the second number : ");

num2 = Convert.ToInt64(Console.ReadLine());

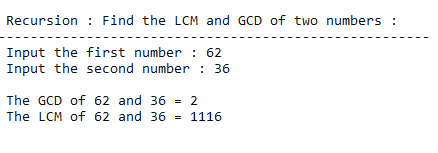
hcf = gcd(num1, num2);

lcm = (num1 \* num2) / hcf;

Console.WriteLine("\n The GCD of {0} and {1} = {2} ", num1, num2, hcf);

Console.WriteLine(" The LCM of {0} and {1} = {2}\n", num1, num2, lcm);

}

**Output:**

**Task No. 6:** Implement file code

**Solution:**

private static Dictionary<string, string> errors = new Dictionary<string, string>();

public static List<FileInfo> result = new List<FileInfo>();

public static void SearchForFiles(string path)

{

DirectoryInfo dirinfo = new DirectoryInfo(path);

try

{

foreach (var fileName in dirinfo.GetFiles())

{

result.Add(fileName);

}

foreach (string directory in Directory.GetDirectories(path))

{

SearchForFiles(directory);

}

}

catch (System.Exception ex)

{

errors.Add(path, ex.Message);//Stores Error Messages in a dictionary with path in key

}

}

static void Main(string[] args)

{

string path = @"C:\Users\mutay\Desktop\Testing";

SearchForFiles(path);

foreach (var item in result)

{

Console.WriteLine(item);

}

}

**Output:**

Text

Description automatically generated

Only File Name

