1)

```
#include<stdio.h>
int  numPrint(int);
int main()
{
    int n = 1;
    printf("\n\n Recursion : print first 50 natural 
numbers :\n");
    printf("-----------------------------------------
---------\n");
    printf(" The natural numbers are :");
    numPrint(n);
    printf("\n\n");
    return 0;
}
int numPrint(int n)
{
    if(n<=50)
    {
         printf(" %d ",n);
         numPrint(n+1);
    }
}
```

2)

```
#include<stdio.h>

int sumOfRange(int);
int main()
{
   int n1;
   int sum;
    printf("\n\n Recursion : calculate the sum of
numbers from 1 to n :\n");
    printf("-----------------------------------------
------------------\n");

   printf(" Input the last number of the range
starting from 1 : ");
   scanf("%d", &n1);

   sum = sumOfRange(n1);
   printf("\n The sum of numbers from 1 to %d :
%d\n\n", n1, sum);

   return (0);
}

int sumOfRange(int n1)
{
   int res;
   if (n1 == 1)
   {
      return (1);
   } else
   {
      res = n1 + sumOfRange(n1 - 1); //calling the
function sumOfRange itself
   }
   return (res);
}
```

3)

```
#include<stdio.h>

int term;
int fibonacci(int prNo, int num);

void main()
{
   static int prNo = 0, num = 1;
    printf("\n\n Recursion : Print Fibonacci Series
:\n");
    printf("----------------------------------------
\n");

   printf(" Input number of terms for the Series (<
20) : ");
   scanf("%d", &term);
 printf(" The Series are :\n");
   printf(" 1  ");
   fibonacci(prNo, num);
    printf("\n\n");
}
```

```
int fibonacci(int prNo, int num)
{
   static int i = 1;
   int nxtNo;

   if (i == term)
      return (0);
   else
   {
      nxtNo = prNo + num;
      prNo = num;
      num = nxtNo;
      printf("%d  ", nxtNo);

      i++;
      fibonacci(prNo, num); //recursion, calling the
function fibonacci itself
   }
   return (0);
}
```

4)

```
#include <stdio.h>
#define MAX 100

void ArrayElement(int arr1[], int st, int l);
```

```
int main()
{
    int arr1[MAX];
    int n, i;
    printf("\n\n Recursion : Print the array elements
:\n");
    printf("-----------------------------------------
--\n");

    printf(" Input the number of elements to be
stored in the array :");
    scanf("%d",&n);

    printf(" Input %d elements in the array :\n",n);
    for(i=0;i<n;i++)
        {
          printf(" element - %d : ",i);
          scanf("%d",&arr1[i]);
        }

    printf(" The elements in the array are : ");
    ArrayElement(arr1, 0, n);//call the function
ArrayElement
    printf("\n\n");
    return 0;
}

void ArrayElement(int arr1[], int st, int l)
{
    if(st >= l)
```

```
        return;

    //Prints the current array element
    printf("%d  ", arr1[st]);

    /* Recursively call ArrayElement to print next
element in the array */
    ArrayElement(arr1, st+1, l);//calling the
function  ArrayElement itself
}
```

5)

```
#include<stdio.h>

int noOfDigits(int n1);
int main()
{
  int n1,ctr;
    printf("\n\n count the digits of a given number
:\n");
    printf("-----------------------------------------
\n");
    printf(" Input  a number : ");
    scanf("%d",&n1);

    ctr = noOfDigits(n1);
    printf(" The number of digits in the number is :
%d \n\n",ctr);
    return 0;
}

int noOfDigits(int n1)
{
    static int ctr=0;

     if(n1!=0)
     {
          ctr++;
         noOfDigits(n1/10);
    }

    return ctr;
}
```

6)

```
#include <stdio.h>

int DigitSum(int num);

int main()
{
    int n1, sum;
    printf("\n\n Recursion : Find the sum of digits
of a number :\n");
```

```
    printf("-----------------------------------------------------------\n");
    printf(" Input any number to find sum of digits: ");
    scanf("%d", &n1);

    sum = DigitSum(n1);//call the function for calculation

    printf(" The Sum of digits of %d = %d\n\n", n1, sum);

    return 0;
}

int DigitSum(int n1)
{
    if(n1 == 0)
        return 0;

    return ((n1 % 10) + DigitSum(n1 / 10));//calling the function DigitSum itself
}
```

7)

```
#include<stdio.h>

int findGCD(int num1,int num2);
int main()
{
  int num1,num2,gcd;
  printf("\n\n Recursion : Find GCD of two numbers
:\n");
  printf("------------------------------------------
\n");
  printf(" Input 1st number: ");
  scanf("%d",&num1);
  printf(" Input 2nd number: ");
  scanf("%d",&num2);

  gcd = findGCD(num1,num2);
  printf("\n The GCD of %d and %d is:
%d\n\n",num1,num2,gcd);
  return 0;
}

int findGCD(int a,int b)
{
     while(a!=b)
     {
          if(a>b)
              return findGCD(a-b,b);
          else
             return findGCD(a,b-a);
     }
     return a;
}
```

8)

```
#include<stdio.h>
#define MAX 100

int MaxElem(int []);
int n;

int main()
{
    int arr1[MAX],hstno,i;
    printf("\n\n Recursion : Get the largest element
of an array :\n");
    printf("----------------------------------------
-------------\n");

      printf(" Input the number of elements to be
stored in the array :");
       scanf("%d",&n);

       printf(" Input %d elements in the array
:\n",n);
       for(i=0;i<n;i++)
        {
          printf(" element - %d : ",i);
          scanf("%d",&arr1[i]);
        }
    hstno=MaxElem(arr1);//call the function MaxElem
to return the largest element
```

```
    printf(" Largest element of the array is:
%d\n\n",hstno);
    return 0;
}
int MaxElem(int arr1[])
{
    static int i=0,hstno =-9999;
    if(i < n)
    {
         if(hstno<arr1[i])
           hstno=arr1[i];
      i++;
      MaxElem(arr1);//calling the function MaxElem
itself to compare with further element
    }
    return hstno;
}
```

9)

```
#include<stdio.h>
#define MAX 100
char* ReverseOfString(char[]);

int main()
{

    char str1[MAX],*revstr;
    printf("\n\n Recursion : Get reverse of a string
:\n");
    printf("----------------------------------------
-\n");

    printf(" Input any string: ");
    scanf("%s",str1);

    revstr = ReverseOfString(str1);//call the
function ReverseOfString

    printf(" The reversed string is: %s\n\n",revstr);
    return 0;
}
char* ReverseOfString(char str1[])
{
    static int i=0;
    static char revstr[MAX];
    if(*str1)
    {
         ReverseOfString(str1+1);//calling the
function ReverseOfString itself
         revstr[i++] = *str1;
    }
    return revstr;
}
```

10)

```
#include<stdio.h>
int findFactorial(int);

int main()
{
  int n1,f;
    printf("\n\n Recursion : Find the Factorial of a
number :\n");
    printf("-----------------------------------------
--------\n");
  printf(" Input  a number : ");
  scanf("%d",&n1);
  f=findFactorial(n1);//call the function
findFactorial for factorial
  printf(" The Factorial of %d is : %d\n\n",n1,f);
  return 0;
}

int findFactorial(int n)
{
   if(n==1)
       return 1;
   else
       return(n*findFactorial(n-1));// calling the
function findFactorial to itself recursively
 }
```

11)

```
#include<stdio.h>

long convertBinary(int);

int main()
{
    long biNo;
    int decNo;

    printf("\n\n Recursion : Convert decimal number
to binary :\n");
    printf("--------------------------------------------
----------\n");

    printf(" Input any decimal number : ");
    scanf("%d",&decNo);

    biNo = convertBinary(decNo);//call the function
convertBinary
    printf(" The Binary value of decimal no. %d is :
%ld\n\n",decNo,biNo);
    return 0;
}
long convertBinary(int decNo)
{
    static long biNo,r,fctor = 1;

    if(decNo != 0)
    {
        r = decNo % 2;
        biNo = biNo + r * fctor;
        fctor = fctor * 10;
        convertBinary(decNo / 2);//calling the 
function convertBinary itself recursively
    }
    return biNo;
}
```

12)

```
#include<stdio.h>

int checkForPrime(int,int);

int main()
{

    int n1,primeNo;

    printf("\n\n Recursion : Check a number is prime 
number or not :\n");
    printf("-----------------------------------------
---------------\n");

    printf(" Input any positive number : ");
    scanf("%d",&n1);

    primeNo = checkForPrime(n1,n1/2);//call the 
function checkForPrime
```

```
   if(primeNo==1)
        printf(" The number %d is a prime number.
\n\n",n1);
   else
      printf(" The number %d is not a prime number.
\nn",n1);
   return 0;
}

int checkForPrime(int n1,int i)
{
    if(i==1)
    {
        return 1;
    }
    else
    {
       if(n1 %i==0)
         return 0;
       else
         checkForPrime(n1,i-1);//calling the function
checkForPrime itself recursively
    }
}
```

13)

```
#include <stdio.h>
```

```
int lcmCalculate(int a, int b);

int main()
{
    int n1, n2, lcmOf;
    printf("\n\n Recursion : Find the LCM of two
numbers :\n");
    printf("----------------------------------------
-----\n");
    printf(" Input 1st number for LCM : ");
    scanf("%d", &n1);
    printf(" Input 2nd number for LCM : ");
    scanf("%d", &n2);
// Ensures that first parameter of lcm must be
smaller than 2nd
    if(n1 >  n2)
        lcmOf = lcmCalculate(n2, n1);//call the
function lcmCalculate for lcm calculation
    else
        lcmOf = lcmCalculate(n1, n2);//call the
function lcmCalculate for lcm calculation
    printf(" The LCM of %d and %d :  %d\n\n", n1, n2,
lcmOf);
    return 0;
}
int lcmCalculate(int a, int b)//the value of n1 and
n2 is passing through a and b
{
    static int m = 0;
    //Increments m by adding max value to it
    m += b;
//  If found a common multiple then return the m.
    if((m % a == 0) && (m % b == 0))
    {
        return m;
    }
    else
    {
        lcmCalculate(a, b);//calling the function
lcmCalculate itself
    }
}
```

14)

```
#include <stdio.h>
void EvenAndOdd(int stVal, int n);

int main()
{
    int n;
    printf("\n\n Recursion : Print even or odd
numbers in a given range :\n");
    printf("-----------------------------------------
--------------------\n");

    printf(" Input the range to print starting from 1
: ");
    scanf("%d", &n);
    printf("\n All even numbers from 1 to %d are : ",
n);
    EvenAndOdd(2, n);//call the function EvenAndOdd
for even numbers

    printf("\n\n All odd numbers from 1 to %d are :
", n);
    EvenAndOdd(1, n);// call the function EvenAndOdd
for odd numbers
    printf("\n\n");

    return 0;
}
void EvenAndOdd(int stVal, int n)
{
    if(stVal > n)
        return;
    printf("%d  ", stVal);
    EvenAndOdd(stVal+2, n);//calling the function
EvenAndOdd itself recursively
}
```

15)

```
#include<stdio.h>
#define MAX 10

void multiplyMatrix(int [MAX][MAX],int [MAX][MAX]);
```

```
int rone,cone,rtwo,ctwo;
int crm[MAX][MAX];

int main()
{

    int arm[MAX][MAX],brm[MAX][MAX],i,j,k;

       printf("\n\n Multiplication of two Matrices
:\n");
       printf("-----------------------------------
\n");

    printf(" Input number of rows for the first
matrix : ");
    scanf("%d",&rone);
    printf(" Input number of columns for the first
matrix : ");
    scanf("%d",&cone);

    printf(" Input number of rows for the second
matrix : ");
    scanf("%d",&rtwo);
    printf(" Input number of columns for the second
matrix : ");
    scanf("%d",&ctwo);

    if(cone!=rtwo)
    {
         printf("\n Check col. of first and row of
second matrix.");
         printf("\n They are different. Try
again.\n");
    }
  else
  {

      printf("\n Input elements in the first matrix
:\n");
      for(i=0;i<rone;i++){
      for(j=0;j<cone;j++){
           printf(" element - [%d],[%d] : ",i,j);
           scanf("%d",&arm[i][j]);}}

      printf(" Input elements in the second matrix
:\n");
      for(i=0;i<rtwo;i++){
      for(j=0;j<ctwo;j++){
           printf(" element - [%d],[%d] : ",i,j);
           scanf("%d",&brm[i][j]);}}

      printf("\n Here is the elements of First matrix
: \n");
      for(i=0;i<rone;i++)
      {
      printf("\n");
      for(j=0;j<cone;j++)
      {
           printf(" %d\t",arm[i][j]);
      }
      }

      printf("\n Here is the elements of Second
matrix : \n");
      for(i=0;i<rtwo;i++)
      {
      printf("\n");
      for(j=0;j<ctwo;j++)
      {
           printf(" %d\t",brm[i][j]);
      }
      }
      multiplyMatrix(arm,brm);
  }

  printf("\n The multiplication of two matrix is :
\n");
  for(i=0;i<rone;i++)
  {
      printf("\n");
      for(j=0;j<ctwo;j++)
      {
           printf(" %d\t",crm[i][j]);
      }
  }
  printf("\n\n");
  return 0;

}
```

```
void multiplyMatrix(int arm[MAX][MAX],int
brm[MAX][MAX])
{

    static int sum,i=0,j=0,k=0;

    if(i<rone)
    { //row of first matrix
    if(j<ctwo)
    {  //column of second matrix
         if(k<cone)
         {
             sum=sum+arm[i][k]*brm[k][j];
             k++;
             multiplyMatrix(arm,brm);
         }
         crm[i][j]=sum;
             sum=0;
             k=0;
             j++;
             multiplyMatrix(arm,brm);
    }
    j=0;
    i++;
    multiplyMatrix(arm,brm);
    }
}
```

16)
```
#include <stdio.h>

#include <string.h>

void checkPalindrome(char [], int);

int main()
{
    char wordPal[25];
    printf("\n\n Recursion : Check a given string is
Palindrome or not :\n");
    printf("------------------------------------------
-----------------\n");

    printf(" Input  a word to check for palindrome :
");
    scanf("%s", wordPal);
    checkPalindrome(wordPal, 0);//call the function
for checking Palindorem

    return 0;
}

void checkPalindrome(char wordPal[], int index)
{
    int len = strlen(wordPal) - (index + 1);
    if (wordPal[index] == wordPal[len])
    {
        if (index + 1 == len || index == len)
        {
            printf(" The entered word is a 
palindrome.\n\n");
            return;
        }
        checkPalindrome(wordPal, index + 1);//calling 
the function itself recursively
    }
    else
    {
        printf(" The entered word is not a 
palindrome.\n\n");
    }
}
```

17)

```
#include <stdio.h>

long int CalcuOfPower(int x,int y)
{
    long int result=1;
    if(y == 0) return result;
    result=x*(CalcuOfPower(x,y-1));  //calling the 
function CalcuOfPower itself recursively
}
int main()
{
    int bNum,pwr;
    long int result;
    printf("\n\n Recursion : Calculate the power of
any number :\n");
    printf("-----------------------------------------
-----------\n");

    printf(" Input the base  value : ");
    scanf("%d",&bNum);

    printf(" Input the value of power : ");
    scanf("%d",&pwr);

    result=CalcuOfPower(bNum,pwr);//called the
function CalcuOfPower

    printf(" The value of %d to the power of %d is :
%ld\n\n",bNum,pwr,result);

    return 0;
}


18) #include <stdio.h>

// function to generate next number
   int getNextValue(int aNum)
   {
      int i = aNum;
      if (i % 2 == 0)
      {
            i = i/2;
        }
        else
        {
            i = 3 * i + 1;
        }
        return (i);// returning the value of next
number to the called function
    }
// function to generate Hailstone number
    void getHailstone(int aNum)
    {
        int hlSe = aNum;
        if (hlSe == 1)
        {
            printf("%i ", hlSe);
        }
        else
        {
            printf(" %i ", hlSe);
            getHailstone(getNextValue(hlSe));// calling
the function itself recursively
        }
    }
// Function to count the length of the Hailstone
sequence
    int countLength(int aNum)
    {
        int hlSe = aNum;
        if(hlSe == 1)
```

```
      {
         return 1;
      }
      else
      {
         return 1+countLength(getNextValue(hlSe));//
calling the function itself recursively
      }

   }

int main(int argu)
{
   int aNum;
    printf("\n\n Recursion : Hailstone Sequence of a
given number upto 1 :\n");
    printf("-----------------------------------------
---------------------\n");
   printf(" Input any number (positive) to start for
Hailstone Sequence : ");
   scanf("%i", &aNum);

   printf("\n The hailstone sequence starting at %i
is : \n", aNum);
   getHailstone(aNum);
   printf("\n\n");
   printf(" The length of the sequence is %i. \n\n",
countLength(aNum));
      return 0;
}
```

19)

```
#include <stdio.h>

void copyString(char [], char [], int);

int main()
{
    char stng1[20], stng2[20];
    printf("\n\n Recursion : Copy One string to 
another :\n");
    printf("-----------------------------------------
----\n");

    printf(" Input the  string to copy : ");
    scanf("%s", stng1);
    copyString(stng1, stng2, 0);
    printf("\n The string successfully copied.\n\n");
    printf(" The first string is : %s\n", stng1);
    printf(" The copied string is : %s\n\n", stng2);
    return 0;
}

void copyString(char stng1[], char stng2[], int ctr)
{
    stng2[ctr] = stng1[ctr];
    if (stng1[ctr] == '\0')
        return;
    copyString(stng1, stng2, ctr + 1);
}
```

20)

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

char checkCapital(char *);

int main()
{
    char str1[20], singLet;

    printf("\n\n Recursion : Find the first capital
letter in a string :\n");
    printf("------------------------------------------
-------------------\n");

    printf(" Input a string to including one or more
capital letters : ");
    scanf("%s", str1);
    singLet = checkCapital(str1);
    if (singLet == 0)
    {
        printf(" There is no capital letter in the
string :  %s.\n", str1);
    }
    else
    {
        printf(" The first capital letter appears in 
the string %s is %c.\n\n", str1, singLet);    }
        return 0;
    }
    char checkCapital(char *str2)
    {
        static int i = 0;
        if (i < strlen(str2))
        {
            if (isupper(str2[i]))
            {
                return str2[i];
            }
            else
            {
                i = i + 1;
                return checkCapital(str2);
            }
        }
        else return 0;
    }
```

21)

```
#include <stdio.h>
int binarySearch(int*, int, int, int, int);

int main()
```

```
{
    int arr1[10], i, n, md, c, low, hg;

    printf("\n\n Recursion : Binary searching :\n");
    printf("-----------------------------------\n");
    printf(" Input the number of elements to store in
the array :");
    scanf("%d", &n);
    printf(" Input %d numbers of elements in the
array in ascending order :\n", n);
    for (i = 0; i < n; i++)
    {
        printf(" element - %d : ", i);
        scanf("%d", &arr1[i]);
    }
    printf(" Input the number to search : ");
    scanf("%d", &md);
    low = 0, hg = n - 1;
    c = binarySearch(arr1, n, md, low, hg);
    if (c == 0)
        printf(" The search number not exists in the
array.\n\n");
    else
        printf(" The search number found in the
array.\n\n");
    return 0;
}

int binarySearch(int arr1[], int n, int md, int low,
int hg)
```

```
{
    int mid, c = 0;
    if (low <= hg)
    {
        mid = (low + hg) / 2;
        if (md == arr1[mid])
        {
            c = 1;
        }
        else if (md < arr1[mid])
        {
            return binarySearch(arr1, n, md, low, mid
- 1);
        }
        else
            return binarySearch(arr1, n, md, mid + 1,
hg);
    }
    else
        return c;
}
```