Recursion :

→ A function is called recursive if a statement in the body of the function calls itself

→ Recursion is the process of definінig something in terms of itself.

→ C functions may be ued recursively ; that is a function may call itself either directly or indirectly

→ If func1() and func2() are two functions, then the direct call structure is ;

```
func1()
  {
   ------
   ------
   func1();
   ------
  } ------
```

i.e function func1() calls itself directly.

and indirect call structure is ;

```
func1()
   {
    -----
    -----
    func2();
    -----
   } -----
func2()
   {
    -----
    -----
    func1() ;
    -----
    -----
}
```

The function func1() calls itself indirectly i.e. func1() calls func2() and func2() once again calls function func1()

Examples of recursion :

```
/* write a program to find the factorial of a number using
   recursion */

# include <stdio.h>
# include < Conio.h>
int fact (int ) ;
void main( )
  {
     int  a, f ;
     clrscr();
    printf (" Enter the number :"),
    scanf ( " %d", &a) ;
  f = fact (a) ;
 printf (" The factorial of %d is = %d\n", a, f) ;
 getch();
}
int fact (int x)
 {
    int f ;
  if ( x = = 1)
     {
      return x ;
     }
  else
     {
       f = x * fact(x-1) ;
     }
    return f ;
 }
```

```
/* write a program to find the power value of a number
      using recursion */

# include <stdio.h>
# include <conio.h>
int  power ( int , int ) ;
void  main()
 {
    int  n, x , p ;
    clrscr() ;
   printf ( " Enter the  base  and  power  :" ) ;
   scanf ( " %d %d" , &n , &x ) ;
    p  =  power ( n, x ) ;
 printf ( "  power  value  is : %d\n" , p ) ;
getch() ;
}
 int  power ( int  a, int  b)
 {
     int  result ;
  if  ( b = = 0)
            {
             return  1 ;
            }
  else
        {
          result  = (a  *  power ( a , b-1)) ;
        }
     return  result ;
 }
```

```
/* write a program to find the GCD of two numbers using
   recursion */

#include <stdio.h>
#include <conio.h>
int gcd (int m , int n);
void main()
 {
    int a, b, c ;
    clrscr();
   printf(" Enter two numbers :");
   scanf("%d%d",&a,&b);
   if ( a<b)
      {
        c = gcd(a,b) ;
      }
   else
       {
          c = gcd(b,a);
       }
   printf("gcd = %d\n",c);
  getch();
 }
 int gcd ( int x , int y)
 {
     int rem ;
     rem = y % x ;
  if ( rem = = 0 )
        return x ;
  else
      {
        return gcd(rem , y) ;
      }
 }
```

/* write a program to find the fibonacci series upto a range using recursion */

```
#include <stdio.h>
# include <conio.h>
void fibo (int );
void main( )
 {
   int z ;
   clrscr();
   printf (" Enter the range :");
    scanf ("%d", &z );
    fibo (z) ;
   getch();
 }
 void fibo (int z)
  {
    static int a=0, b=1;
     int c ;
  if (z < 2)
        printf ("%d", a );
  else
        {
           fibo (z-1);
           c = b ;
           b = a+b;
           a = c ;
         printf ("%d", a) ;
        }
  }
```

```
/* program to find out the LCM and HCF of two numbers
     recursively */
```

Least Common multiple

Highest common factor

```
#include <stdio.h>
# include <conio.h>
int hcf (int a, int b);
int lcm (int a, int b);
int m, n,
void main()
{
   int x, y ;
   clrscr();
  printf("Enter two numbers:");
  scanf("%d %d", &x, &y);
 printf(" HCF of %d and %d is %d\n", x, y, hcf(x,y));
  m = x ;
  n = y;
 printf("LCM of %d and %d is %d\n", x, y, lcm(x,y));
 getch();
}
int hcf ( int a, int b)
  {  if (a==b)
         return b;
 else if (a<b)
       hcf (a, b-a);
 els
      hcf (a-b, b);
 }
int  lcm (int a, int b)
      {
        if (a==b)
          return b;
else  if (a<b)
    lcm (a+m, b);
else
      lcm (a. b+n);
}

/* write a program to print the reverse of a positive number
   using recursion */
# include <stdio.h>
# include <conio.h>
void  reverse  (long int n);
void  main()
 {
    long int num;
    clrscr();
   printf ("Enter the number :");
   scanf ("%d", &num);
     reverse (num);
   getch();
 }
 void  reverse  (long int n)
  {   int  rem ;
     if (n==0)
          return ;
  else
       {
          rem = n % 10;
          printf ("%d", rem);
           n = n/10;
        reverse (n);
     }
  }
```