Functions in 'C' programming :

A function is a block of code that performs a specific task. It has a name and it is reusable in 'C' program as required. It also optionally returns a value to the calling program. A function has some properties. These are:

(i) Every function has a unique name. This name is used to call function from main() function. A function can also be called from within another function.

(ii) A function is independent and it can perform its task without intervention from or interfering with other parts of the program.

(iii) A function returns a value to the calling program. This is optional and depends upon the task your function is going to accomplish.

(iv) It facilitates top-down modular programming. In this programming style, the high level logic of the overall problem is solved first while the details of each lower-level function are addressed later.

(v) The length of a source program can be reduced by using functions at appropriate places.

(vi) It is easy to locate and isolate a faulty function for further investigations.

15-04-2011

Structure of a Function:

Appointments ~ Meetings

A general form of a 'C' function looks like this:

```
<Return type> Function_Name (Argument 1, Argument 2, ---)
{
    statement 1;
    statement 2;
    statement 3;
}
```

An Example of a Function

```
void add (int x, int y)
{
    int z;
    z = x + y;
    printf(" %d ", z);
}
```

Elements of User-defined Function

In order to make use of a user-defined function,

We need to establish three elements that are related to functions:

(i) Function Declaration

(ii) Function call

(iii) Function Definition.

(i) Function Declaration :- The program or a function that calls the function is referred to as the calling program or calling function. The calling program should declare any function like declaration of a variable that is to be used later in the program. This is known as the function declaration or function prototype.

(ii) Function Call :- In order to use the user-defined function, we need to invoke it at a required place in the program. This is known as the function call.

(iii) Function Definitions:-

The function definition is an independent program module that is specially written to implement the requirements of the function.

Syntax of User-Defined Function with a program:

```
#include <stdio.h>
#include < conio.h>
void  add ( int x, int y) ; → Function Declaration
void main ()
{
  - - — —
   - ——
   - ——
   ————
   add ( a , b ) ;        ————→ Function calling
    - - — ——
    - ———
    - ———
 }

void ( int a , int b)  }
{                      }
    - - — ——           }
    - ——               }
    -  ——              } ——→ Function Definition.
    - ——               }
     ———               }
  }
```

Types of User-Defined Functions

Appointments ~ Meetings

Mainly, There are five types of User-Defined Functions. These are:

(i) Functions with no arguments and no return values.

(ii) Functions with arguments and no return values.

(iii) Functions with arguments and one return value.

(iv) Functions with no arguments but return a value.

(v) Functions that return multiple values.

(i) Functions with no arguments and no return values.

Syntax:

```
void  add ( ) ;  ⟶  Fun. Declaration
void main()
{
    - ———
    —— —
    ————
    - —  ——
    add ( ) ;      ⟶  Fun. Calling
    ————
    —  —
}

void  add ( )   }
{               }
  - - ———       }  ⟶  Fun. Definition.
  ——  —         }
  -  ———        }
  —  ——         }
}
```

It is the one of the type of user-defined function. It does not receive any data from the calling function. It does not return any value to the calling function, i.e the calling function does not receive any data from the called function. In effect, There is no data transfer between the calling function and the called function.

22-04-2011

Example : 1 : Write a 'C' program to print the addition of two nos by using function with no arguments and no return values.

```
#include<stdio.h>
#include<conio.h>
void add();
void main()
{
  clrscr();
  add();
  getch();
 }
void add()
{
  int x, y, z;
  printf("Enter two Nos");
  scanf("%d%d", &x, &y);
  z = x+y;
  printf("%d", z);

 }
```

(ii) Function with arguments and no return values:

Syntax:

```
void add ( int , int ); ⟶ Fun Declaration
void main ()
{
  - - - -
  - - - -
  - - - -

  add ( a , b ) ;   ⟶ Fun. Calling
  ________
  ________

  getch ();
}

void add ( int a , int b ) }
{                          }
  ________                 }
  - ______                 }⟶ Fun Definition.
  - ______                 }
  - ______                 }
}
```

In this type of function, calling function sends some value as an argument to the called function and called function does not return any value to the calling function.

Example: Write a 'C' program to print the addition of two nos. by using function with arguments and no return values.

```
#include<stdio.h>
#include< conio.h>
void add( int, int);
void main()
{
   int a, b;
   clrscr();
   printf(" Enter two nos.");
   scanf("%d%d", &a, &b);
   add(a, b);
   getch();
 }
void add( int a, int b)
{
   int c;

   c = a+b;
   printf(" %d", c);

 }
```

Q. Write a C program to print the addition and subtraction of two no. by using function with arguments and no return values.

```
#include<stdio.h>
#include<conio.h>
void arithmetic(int, int);
void main()
{
   int a,b;
   clrscr();
   printf(" Enter two nos");
   scanf("%d%d", &a,&b);
   add(a,b);
   getch();
}

void arithmetic(int a, int b)
{
    int c,d;
    c=a+b;
    d=a-b;
    printf("%d%d", c,d);
}
```

M	T	W	T	F	S	S
30	31	•	•	•	•	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

MAY

Q: Write a C program to print the area and perimeter of a circle by using function with arguments and no return values.

Appointments Meetings

```
#include <stdio.h>
#include <conio.h>
void circle(int);
void main()
{
  int r;
  clrscr();
  printf(" Enter radius of a cirle");
  scanf("%d", &r);
  circle(r);
  getch();
}

void circle(int r)
{
   float a, p;

   a = 3.14 * r * r;
   p = 2 * 3.14 * r;
   printf(" %f %f", a, p);
}
```

Q2: Print the area and perimeter of a rectangle by using fun. with arguments and no return values.

```
#include<stdio.h>
#include<conio.h>
void rectangle(int,int);
void main()
{
        int l,b;
        clrscr();
        printf("Enter length and breadth of rectangle");
        scanf("%d%d",&l,&b);
        rectangle(l,b);
        getch();
}

void rectangle(int l, int b)
{
        int a,p;
        a=l*b;
        p=2*(l+b);
        printf("%d%d",a,p);
}
```

Q2: Print the addition, subtraction, multiplication, division and modulus of two numbers by using function with arguments and no return values.

```
#include<stdio.h>
#include<conio.h>
void arithmetic(int,int);
void main()
{
        int x,y;
        clrscr();
        printf("Enter two nos");
        scanf("%d%d",&x,&y);
        arithmetic(x,y);
        getch();
}
void arithmetic(int x, int y)
{
        int a,s,m,d,mo;
        a=a+b;
        s=a-b;
        m=a*b;
        d=a/b;
        mo=a%b;
        printf("%d%d%d%d%d",a,s,m,d,mo);
}
```

(iii) Function with arguments and one return value :

Syntax:

```
int add ( int , int );  ——→ Fun. Declaration
int main ( )
{
  - -
  - —
  - —

  x = add ( a , b );  ——→ Fun. calling
  ————————

  return 0;
}

int add ( int , int )   }
{                       }
  ———                   }——→ Fun. Definition.
  —   —                 }
  —   —                 }
  —   —                 }
  return z;             }
}
```

Example: Write a 'C' program to print the addition of two nos. by using function with arguments and one return value.

```
#include <stdio.h>
#include < conio.h>
int  add ( int, int) ;
int main ()
{
    int  x, a, b ;
    clrscr();
    printf ("  Enter two nos");
    scanf(" %d%d ", &a, &b);

     x = add(a, b);
     printf(" %d", x);
     getch()
     return 0;
  }

  int add (int a, int b)
  {
      int z;
      z = a+b;
      return z;
   }
```

Q10: print the area and perimeter of a triangle by using function with arguments and one return value.

```
#include<stdio.h>
#include<conio.h>
int triangle(int,int,int);
int main()
{
      int area,a,b,c;
      clrscr();
      printf("Enter height,base and perpendicular of a
                   right angle triangle");
      scanf("%d%d%d",&a,&b,&c);
      area=triangle(a,b,c);
      printf("Area = %d",area);
      getch();
      return 0;
}
int triangle(int a,int b, int c)
{
      int z,p;
      z=(a*b)/2;
      p=a+b+c;
      printf("Perimeter = %d",p);
      return z;
}
```

(iv) Functions with no arguments but return a value:

Syntax:

```
int add( ); ⟶ Fun. Declaration
int main()
{
  — —
  — —
  ——
   x = add( ); ⟶ Fun. Calling
   ————
   ————
 }

 int add( )  ⎫
 {           ⎪
   - ———     ⎪
   - ———     ⎬⟶ Fun. Definition
   - ———     ⎪
   return z; ⎪
 }           ⎭
```

Example : Write a 'C' program to print the addition of two nos, by using Fun. with no arguments but return a value.

```
#include <stdio.h>
#include <conio.h>
int add ( );
int main ( )
{
    int x ;
    clrscr()
    x = add ( );
    printf(" %d ", x) ;
    getch();
    return 0;
}

  int add ( )
  {
     int a, b, z ;
     printf(" Enter two Nos");
     scanf( " %d %d ", &a, &b);
     z = a+b;
     return z;
  }
```

Q9: Print the area and perimeter of a circle
    by using function with no arguments but return a value.

```
#include<stdio.h>
#include<conio.h>
float circle();
int main()
{
      float a;
      clrscr();
      a=circle();
      printf("Area = %f",a);
      getch();
      return 0;
}
float circle()
{
      int r;
      float z,p;
      printf("Enter radius of a circle");
      scanf("%d",&r);
      z=3.14*r*r;
      p=2*3.14*r;
      printf("Perimeter = %f",p);
      return z;
}
```

Q8: print the area and perimeter of a triangle by using function with no arguments but return a value.

```
#include<stdio.h>
#include<conio.h>
int triangle();
int main()
{
      int a;
      clrscr();
      a=triangle();
      printf("Area = %d",a);
      getch();
      return 0;
}
int triangle()
{
      int a,b,c,z,p;
      printf("Enter height,base and perpendicular of a
                    right angle triangle");
      scanf("%d%d%d",&a,&b,&c);
      z=(a*b)/2;
      p=a+b+c;
      printf("Perimeter = %d",p);
      return z;
}
```

(v) Functions that return multiple values :

Syntax :

```
void mathoperation ( int , int , int * , int * );

void main()
 {
   __________
   __________
   __________
   mathoperation( x, y, &a, &b );
   __________
   __________
 }

void mathoperation( int , int , int * , int * )
 {
   __________
   __________
   __________
 }
```

M	T	W	T	F	S	S
30	31	•	•	•	•	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

MAY

Q1: Write a 'C' program to print the addition and Subtraction of two numbers by using functions that return multiple values.

Appointments ~ Meetings

```
#include< stdio.h>
# include < conio.h>
void mathoperation( int, int, int *, int *);
void main()
{
    int x, y, s, d;
    clrscr();
    printf(" Enter two nos");
    scanf("%d%d", &x, &y);
    mathoperation(x, y, &s, &d);
    printf("%d %d", s, d);
    getch();

}

void mathoperation( int x, int y, int *s, int *d)
{


    *s = x+y;
    *d = x-y;

}
```