Return values and their types

- In C, function may or may not return value, from called-function to calling-function.
- If it returns value, it is done through the return-statement.
- Calling-function can pass n-number of parameters to the called-function.
- But called function can only return one value.
- The return statement can take one of the following forms :-

`return;` or

`return (expression);`

- The first plain 'return' does not return any value, it acts as a closing brace of the function.
- When 'return' is encountered, the control is immediately passed back to the calling function.

ex. Example for simple 'return' as below:-

```
if (error)
    return;
```

- The second form of 'return with expression' return the value of the expression.

ex.
```
int add (int x, int y)
{
    int z;
    z = x + y;
    return (z);
}
```

- These two statements may be combine & written as below :-

```
return (x*y);
```

→ A function may have more than one return statements. F
This situation arises when, the value returned is based on certain condition

ex.
```
if (x<=0)
    return (0);
else
    return (1);
```

- All the functions by default returns int type data.
- But, if the function returns other type of data there is need to specify the return-type of function, as data-type of return value.

Ex. return-type

```
float add ( )
{
   return (1.5 + 2.3);
}
```

Final sum is 3.8, which is a float value.

Hence, the return-type of function is float

Ex. return-type

```
int add ( )
{
   return (100 + 20);
}
```

Final sum is 120, which is an integer value

Hence, the return-type is int.

* **Advantages of function :-**

Following are the various advantages of user defined function

1) Due to user defined function length of the main program can be minimized.
2) Due to minimized length execution speed will be increases.
3) Due to minimized length programme requires less memory.
4) Due to requires less memory & more execution speed program efficiency will be increases.
5) Program debugging becomes easier.
6) Program testing becomes easier.
7) A user defined function can be placed in a file and can be called by the multiuser in their program.

* **Categories of function :-** or

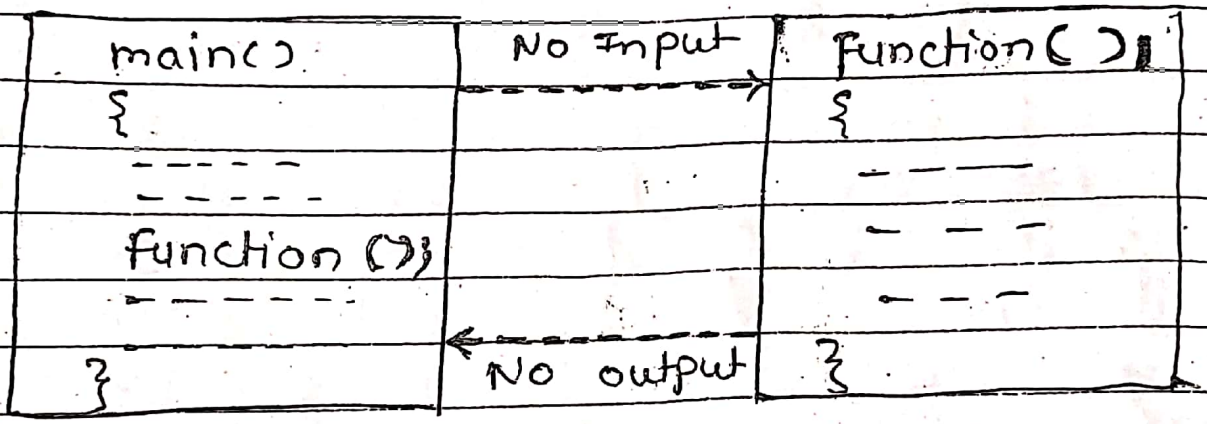
Depending on the actual arguments and Return value user defined function is devided into three categories.

1) function with no argument and No Return value.
2) function with argument and No Return value
3) Function with argument and Return value.
4) Function with No argument and Return value

1) Function with No Argument and No Return value :-

① When a user defined function doer not pass actual Argument from the calling function to the formal argument of called function ② Similarly called function does not return any value to the calling function then such category is called function without argument and no Return value.

The diagramatic representation for this categorie is as follows :-



③ Here dotted line indicates their is no data transmission from the calling function to called function ④ and no return value from called function to calling function.

⑤ In this categorie their is only transfer the control from calling function to called function and called function to calling function.

Ex :- #include <stdio.h>
main()
{
city();
}

function with no parameter

```
city()
{
printf("In My city is Amravati");
}
```

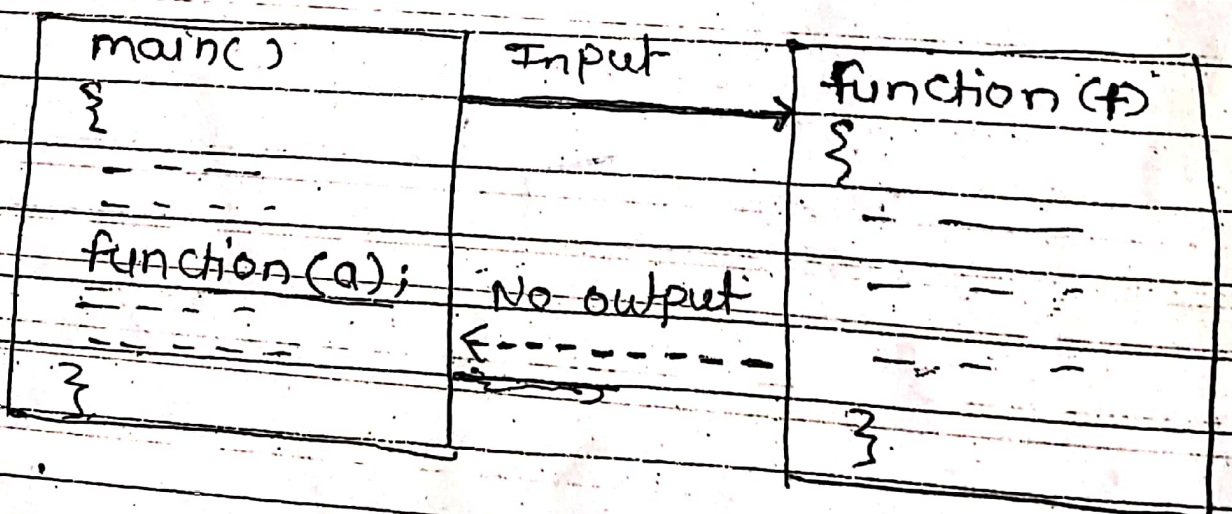
No return value

/* function definition */

2) Function with Argument and No return value :-

① When a user defined function passes actual argument from the calling function to the formal argument of called function, ② and called function does not return any value to the calling function then such category is called function with argument and No Return value.

The diagrametic representation for this categorie is as follows :-



(3) Here bold line indicates their is data transmission from calling function to called function (4) but no return value from called function to calling function. (5) The actual arguments passes its value to the formal arguments. (6) In this category their is tranfer of data from calling function to called function. But only transfer of control from called function to calling function.

:- /* Example of user defined function */

```
#include<stdio.h>
#include<conio.h>
void main()
{
  int a,b; void add(int x, int y);
  printf("\n Enter any two values");
  scanf("%d%d", &a, &b);
  add(a,b); ← function with parameters
  getch();
}
void add(int x, int y)
  {
    int z;
    z = x+y;
    printf("\n sum=%d", z);
  }
```

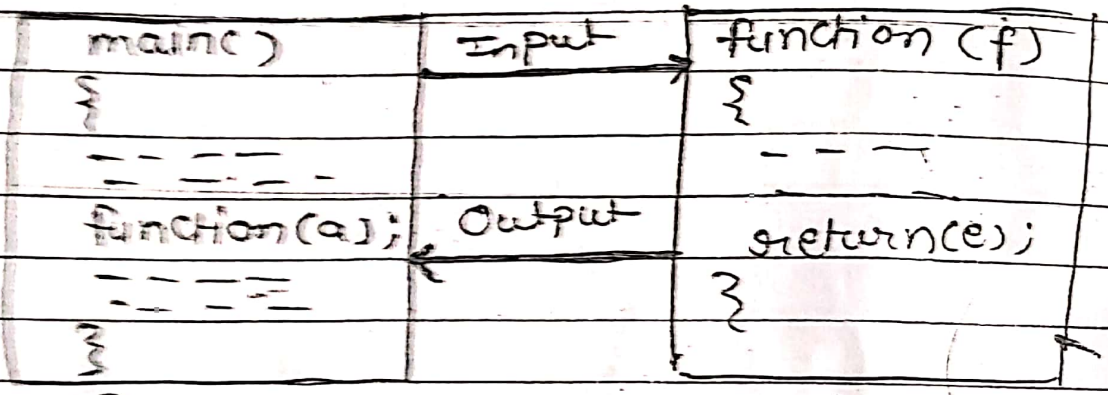
No return value

4) Function with Argument and Return value :-

(1) When a user defined function passes actual argument from the calling function to the formal argument of called function

(2) Similarly called function return value to the calling function then such category is called 'function with Argument and Return value'.

The diagramatic representation for this categorie is as follows :-



(3) Here bold line indicates their is data transmission from calling function to called function and return value from called function to calling function (4) The actual arguments passes its value to the formal arguments (5) In this categorie their is transfer of data from calling function to called function and from called function to calling function.

Ex:- /* Example of user defined function */

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

```
void main()
{
int a,b, add (int, int);
printf("\n Enter any two values");
scanf("%d %d", &a, &b);
c=add (a,b);
printf("\n sum= %d", c);
getch();
}
```

function with parameter

```
int add (int x, int y)
{
  int z;
  z = x+y;
  return z;
}
```

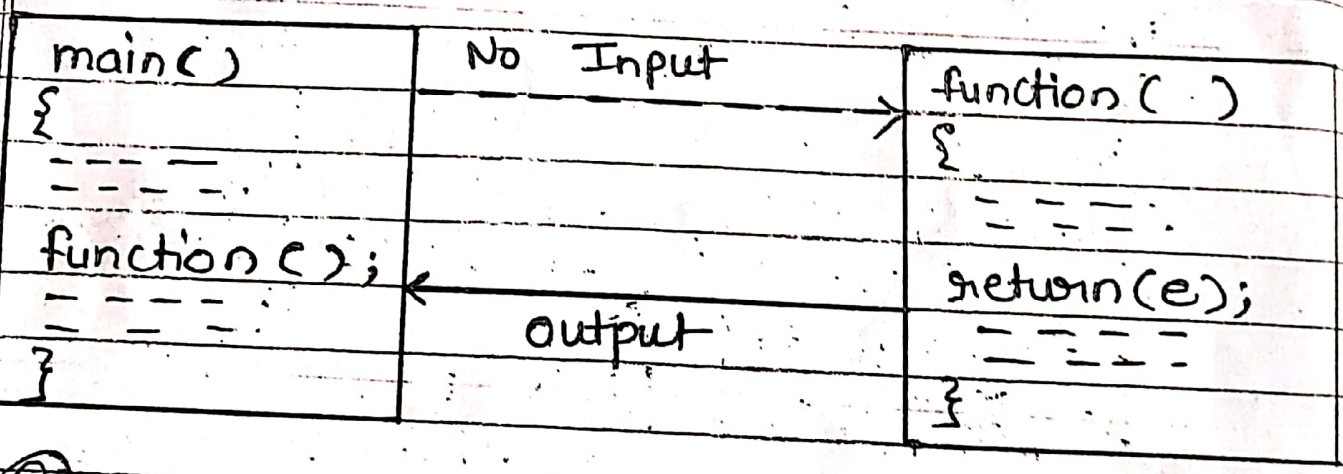
function with return value

4) Function with no argument and Return-Value :-

(1) When a user defined function does not pass, actual argument from the calling function ; to the formal argument of called function.

(2) But called function returns value to the calling function, then such category is called as "function with no argument and return-value".

The diagramatic representation for this category is as follows :-



(3) Here, dotted line indicates their is no data transmission from the calling function to called function, but return-value from called function to calling function.

(4) In this category their is only transfer of the control from calling function to called function, but transfer of data from called function to calling function.

```
/* Example of user defined function */
#include<stdio.h>
#include<conio.h>
void main()
{
   int c;
   int add();
   c = add();
   printf("Sum = %d", c);
   getch();
}

int add()
{
   int x,y,z;
   printf("Enter any two values");
   scanf("%d %d", &x, &y);
   z=x+y;
   return z;
}
```

Calling function with no actual-parameter

Called function with return-value

Unit-I (Questions)

1. Explain following with example:-
   - Function declaration
   - Function definition
   - Function prototype
   - Function calling
   - Function returning
2. What is function? Explain function prototype with suitable example.
3. What is function? Write its format and explain the term function calling and function returning.
4. Explain with example calling function and called fuction.
5. Define the function? Explain with example how function works.
6. Explain with example:-
   - Function with no argument and no return value
   - Function with no argument and return value
   - Function with argument and no return value
   - Function with argument and return value
7. Explain the concept of return values and their types.
8. Explain the categories of function with suitable example.
9. Explain function recursion? Write a program to calculate factorial of a given number using recursion.
10. Explain in detail with suitable example function recursion.
11. Write a program to calculate factorial of a given number using recursion.
12. Explain the concept of function with array.
13. Explain pointer to function with suitable example.
14. Explain the concept of call by value with suitable example.
15. Explain the concept of call by pointer(reference) with suitable example.
16. Explain the concept of call by value and call by reference with suitable example.
17. Differentiate between
    - Call by value and call by reference(pointer)
    - Local variable and global variable
    - Library function and user defined function