C Programming

Trainer : Smita Kadam

Email ID : smita@sunbeaminfo.com

Function

- Set of instructions collected as block and identified with some name as subprogram to complete a specific task.

- Subprogram can be written using
  - Procedure – C language do not support
  - Function
- **Types of Functions**
  - Built-In-Functions (Library Functions)
  - User Defined Functions

Points to Note:

1. Function may or may not take argument
2. Function may or may not return value. However function can return only one value at a time.
3. Helps to re use code.
4. Makes code less complex
5. We can compile code once and use it many times.

Arguments to Function

Can be

1. passed by value : makes a another copy of actual argument into formal argument. Changes inside function modifies local copy of formal parameter hence actual copy of calling function remain unchanged.

2. passed by address : Formal parameter can access directly memory location of calling function. Hence Called function can modify directly memory of calling function.

Points to when you write User Defined Function

Declaration:

<return type> <identifier> ([<argument type>,…]);

Definition:

<return type> <identifier> ([<argument type> <identifier>,…])
{
<statements>
}

Call to function:

<location> = <identifier>([<argument value/address>,…]);

s1

Declaration , Definition , Call

```
int sum(int,int); OR int sum(int n1,int n2);

int sum(int n1, int n2) //formal parameters
{
        return n1 + n2;
}//called function

int main()
{
    int a=23, b=24;
    int ans = sum(a,b);          //actual arguments are passed by value here
    printf("Answer = %d",ans",ans);
}//calling function
```

Declaration of a Function

Definition of a Function

Call of a Function

Slide 5

s1 sunbeam, 01-03-2022

Thank you!