Tutorial Link https://codequotient.com/tutorials/Scope of a function/59fdd5c8e63d6b7fd5debfaf

TUTORIAL

Scope of a function

Chapter

1. Scope of a function

Each function is a block of code. Thus a function defines a block scope. So the body of the function is invisible to the other part of code. The only way to access a function is by calling it. The values passed at the time of calling a function are called actual arguments, which will be resolved with formal arguments of the function, and if everything matches then the execution of current block is suspended and execution of function begins. Following program will demonstrate the function blocks.

```c
#include <stdio.h>
int sum(int m, int n)
{
  printf ("value of m in sum() = %d\n",  m);
  printf ("value of n in sum() = %d\n",  n);
  return m+n;
}

int main ()
{
  int a = 10, b = 20, c = 0;
  printf ("value of a,b in main() = %d \t %d\n",  a, b);
  c = sum( a, b);
  printf ("sum of a and b = %d\n",  c);
  // printf ("value of m in main() = %d \n", m);
  /* Compile time error as m ends as the sum function ends. No variable m exists in function main().*/
  return 0;
```

```
18 }
19
```

The output of the above function is: -

```
value of a,b in main() = 10      20
value of m in sum() = 10
value of n in sum() = 20
sum of a and b = 30
```

If the name of actual parameters and formal parameters are same, it does not mean that they will refer to same memory, each function will have its own copy of variables.