Global namespace scope - the outermost namespace scope of a program, in which objects, functions, types and templates can be defined. A name has global namespace scope if the identifier's declaration appears outside of all blocks, namespaces, and classes.

// Global variable declaration

int x = 6;

void test();

int main()

{

// Outputs 13

cout << x <<endl;

test();

return 0;

}

void test()

{

// Outputs 14

cout << x;

}

Local scope- identifiers declared within a block assume a local scope, They can only be accessed from within the function.

void test();

int main()

{

// local variable to main()

int var = 9;

test();

var1 = 7;

}

void test()

{

// local variable to test()

int var1;

var1 = 4;

cout << var;

}

Function-prototype scope- parameters are visible until the function declaration can be considered closed. Meaning, that you cannot have two parameters with the same name.

**Example: double myTriangle (double base, double height);**

***//base and height have function-prototype scope***

Function scope- When a variable is declared inside a function, it is only accessible within that function and cannot be used outside that function.

function myFunction() {  
  var carName = "Volvo";   // Function Scope  
}