Scope of Variable

Let's classify variables according to their accessibility in the program.

We'll cover the following

Introduction

The scope of a variable defines in which part of the program that particular variable is accessible. In C++, the variable can be

- Local variable
- Global variable

Local variable

Suppose you are staying at a hotel. The hotel manager gives you a key to room No. 5. You can only access room No. 5 in the hotel.

The local variable is just like the hotel room-specific key. It is only accessible within the block in which it is declared.

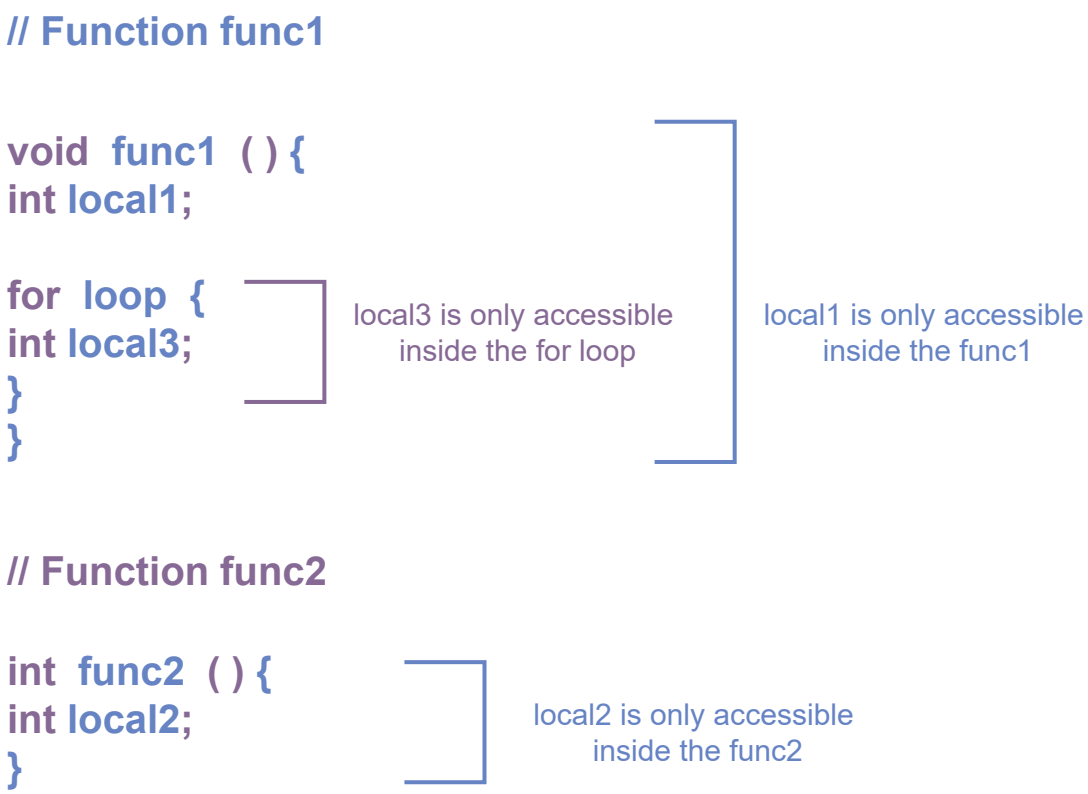
> Block is a section of code enclosed inside the curly braces.

*The **local variable** can only be accessed within the block in which it is declared.*

A block can be a function, loop, or conditional statement. These variables are created when the compiler executes that particular block and destroyed when the compiler exits that block.

Illustration

See the illustration given below!



In the figure above, `local1`, `local2`, and `local3` are the local variables. They are only accessible within the block in which they are declared. We cannot access them outside the block in which they are declared. For example, `local3` is not accessible in `func2( )`.

Example program

RUN the program and see what happens!

> The program will not compile.

```cpp
#include <iostream>
using namespace std;

void function () {
  int function_local = 10;
  cout << main_local;
}

int main() {
  int main_local = 20;
  cout << function_local;
  return 0;
}
```

Explanation

- In the code above, variable `function_local` is only accessible within the body of the `function()`. We cannot access it in the `main()`.
- Similarly, `main_local` is only accessible within the body of the `main()`. We cannot access it in the `function()`.
- The program is generating an error because we are trying to access the variable `main_local` in the body of the `function()` and `function_local` in the body of the `main()`.

Global variable

Again consider the example of a **hotel**. The hotel manager has the master key. Unlike us, the hotel manager can access each and every room in the hotel.

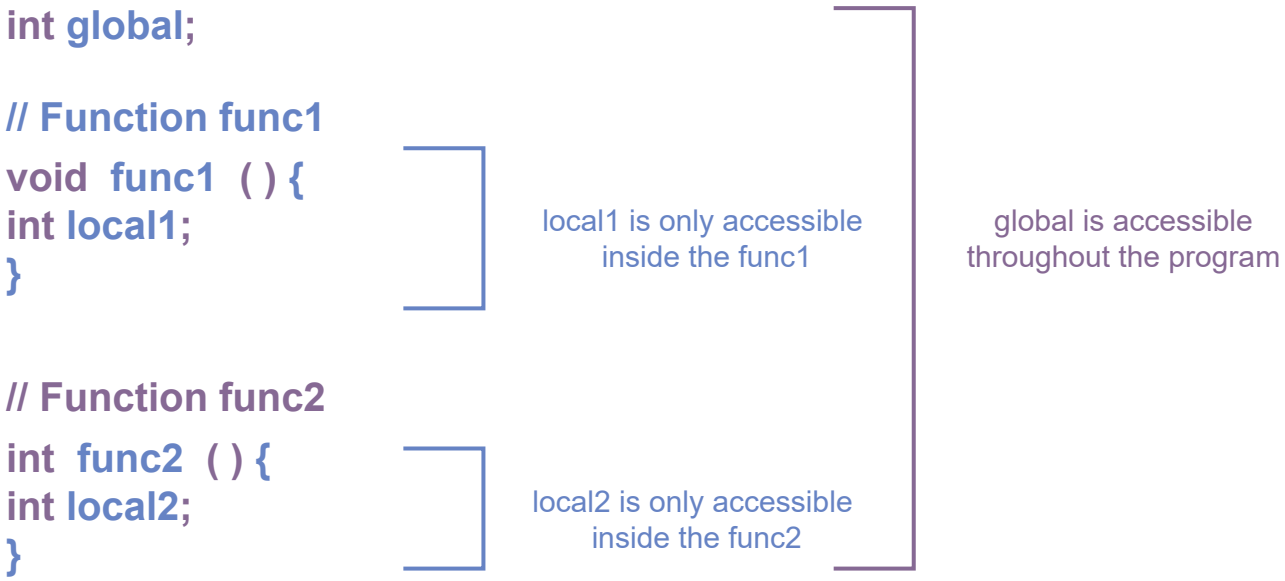
Similar to the master key, global variables are accessible in the whole program.

---

*The **global variables** can be accessed from the point it is declared to the end of the program. They are declared at the very start of the program before defining any function.*

Illustration

See the illustration given below!



In the above illustration, `global` is declared in the start of the program. Therefore, we can access it anywhere in the program. We can access the `global` in `func1()` and `func2()`.

Example program

Press the **RUN** button and see the output!

```
#include <iostream>
using namespace std;
int global = 3;

void function () {
  int function_local = 10;
  cout << "global = " << global << endl;
}
int main() {
  int main_local = 20;
  cout << "global = " << global << endl;
  function();
  return 0;
}
```

Explanation

The variable `global` is declared before any function. Therefore, we can access it from any point in the program. In the above program, the value of global is accessible in both `main()` and `function()`.

Q What is the output of the following code?

```
void function1 () {
  cout << "global = " << global << endl;
}

int global = 3;

void function2 () {
  cout << "global = " << global << endl;
}
int main() {
  cout << "global = " << global << endl;
  function1();
  function2();
```

Retake Quiz

In the upcoming lesson, we will discuss the global variables in more detail.