4.1 — Blocks (compound statements)

BY ALEX ON JUNE 18TH, 2007 | LAST MODIFIED BY ALEX ON SEPTEMBER 9TH, 2017

Blocks (compound statements)

A **block** of statements, also called a **compound statement**, is a group of statements that is treated by the compiler as if it were a single statement. Blocks begin with a { symbol, end with a } symbol, and the statements to be executed are placed in between. Blocks can be used any place where a single statement is allowed. No semicolon is needed at the end of a block.

You have already seen an example of blocks when writing functions:

```
int add(int x, int y)
{ // start a block
    return x + y;
} // end a block

int main()
{ // start a block

    // multiple statements
    int value(0);
    add(3, 4);

    return 0;

} // end a block (no semicolon)
```

Blocks can be nested inside of other blocks. As you have seen, the *if statement* executes a single statement if the condition is true. However, because blocks can be used anywhere a single statement can, we can instead use a nested block of statements to make the *if statement* execute multiple statements if the condition is true!

```
#include <iostream>

int main()
{
    std::cout << "Enter an integer: ";
    int value;
    std::cin >> value;

    if (value >= 0)
    { // start of nested block
        std::cout << value << " is a positive integer (or zero)" << std::endl;
        std::cout << "Double this number is " << value * 2 << std::endl;
    } // end of nested block
    else
    { // start of another nested block
        std::cout << value << " is a negative integer" << std::endl;
        std::cout << "The positive of this number is " << -value << std::endl;
    } // end of another nested block

    return 0;
}
```

If the users enters the number 3, this program prints:

```
Enter an integer: 3
3 is a positive integer (or zero)
Double this number is 6
```

If the user enters the number -4, this program prints:

```
Enter an integer: -4
-4 is a negative integer
The positive of this number is 4
```

It is even possible to put blocks inside of blocks inside of blocks:

```cpp
int main()
{
    std::cout << "Enter an integer: ";
    int value;
    std::cin >> value;

    if (value >  0)
    {
        if ((value % 2) == 0)
        {
            std::cout << value << " is positive and even" << std::endl;
        }
        else
        {
            std::cout << value << " is positive and odd" << std::endl;
        }
    }

    return 0;
}
```

There is no practical limit to how many nested blocks you can have. However, it is generally a good idea to try to keep the number of nested blocks to at most 3 (maybe 4) blocks deep. If your function has a need for more, it's probably time to break your function into multiple smaller functions!

Summary

Blocks allow multiple statements to be used wherever a single statement can normally be used. They are extremely useful when you need a set of statements to execute together.

4.1a -- Local variables, scope, and duration

Index

3.x -- Chapter 3 comprehensive quiz

Share this:

Facebook Twitter Google Pinterest

C++ TUTORIAL | PRINT THIS POST

38 comments to 4.1 — Blocks (compound statements)

Aditi
June 22, 2018 at 8:16 pm · Reply