# 【C++】Dau13(3)

| | |
|---|---|
| Class | C++ |
| Date | @December 1, 2021 |
| Material | |
| Series Number | |
| Summary | |

## 【Ch5】Statements

Statements are executed sequentially. However, for complicated programs, sequential execution is inadequate. Therefore, C++ also defines a set of flow-of-control statements that allow more complicated execution paths.

### 5.1 Simple Statements

*Null Statements*

The simplest statement is the empty statement, also known as a null statement. A null statement is a single semicolon:

```
; //null statement
```

A null statement is useful where the language requires a statement but the program's logic does not. Such usage is most common when a loop's work can be done within its condition.

For example, we might want to read an input stream, ignoring everything we read until we encounter a parciular value|:

```
//read until we hit end-of-file or find an input equal to sough
while (cin >> s && s != sought)
  ; //null statement
```

*Best Practices: Null statements should be commented. That way anyone reading the code can see that the statement was omitted intentionally.*

### *Compound Statements(Blocks)*

A compound statement, usually referred to as a block, is a (possibly empty) sequence of statements and declarations surrounded by a pair of curly braces.

A block is a cope. Names introduced inside a block are accessible only in that block and in blocks nested inside that block. Names are visible from where they are defined until the end of the (immediately) enclosing block.

Compound Statements are used when the language requires a single statement but the logic of our program needs more than one.

For example, the body of a while or for loop must be a single statement, yet we often need to execute more than one statement in the body of a loop. We do so by enclosing the statements in curly braces, thus turning the sequence of statements into a block.

```
while (val M<= 10) {
  sum += val; //assigns sum + val to sum
  ++val; //add 1 to val
}
```

*Note: A block is not terminated by a semicolo*