Chapter 3

Structure of a C++ Program

Brooks/Cole
Thomson Learning™

OBJECTIVES

After studying this chapter you will be able to:

- ❑ Understand the concept of and use expressions.
- ❑ Identify the seven types of C++ expressions.
- ❑ Use basic expressions in a program.
- ❑ Assign expression values to a variable.
- ❑ Evaluate expressions using precedence and associativity.
- ❑ Understand and use expression side effects.
- ❑ Understand and use compound statements.
- ❑ Understand that good functions are simple and short (KISS).
- ❑ Use parentheses to clarify code.
- ❑ Communicate clearly with the user through well written prompts.

Brooks/Cole
Thomson Learning™

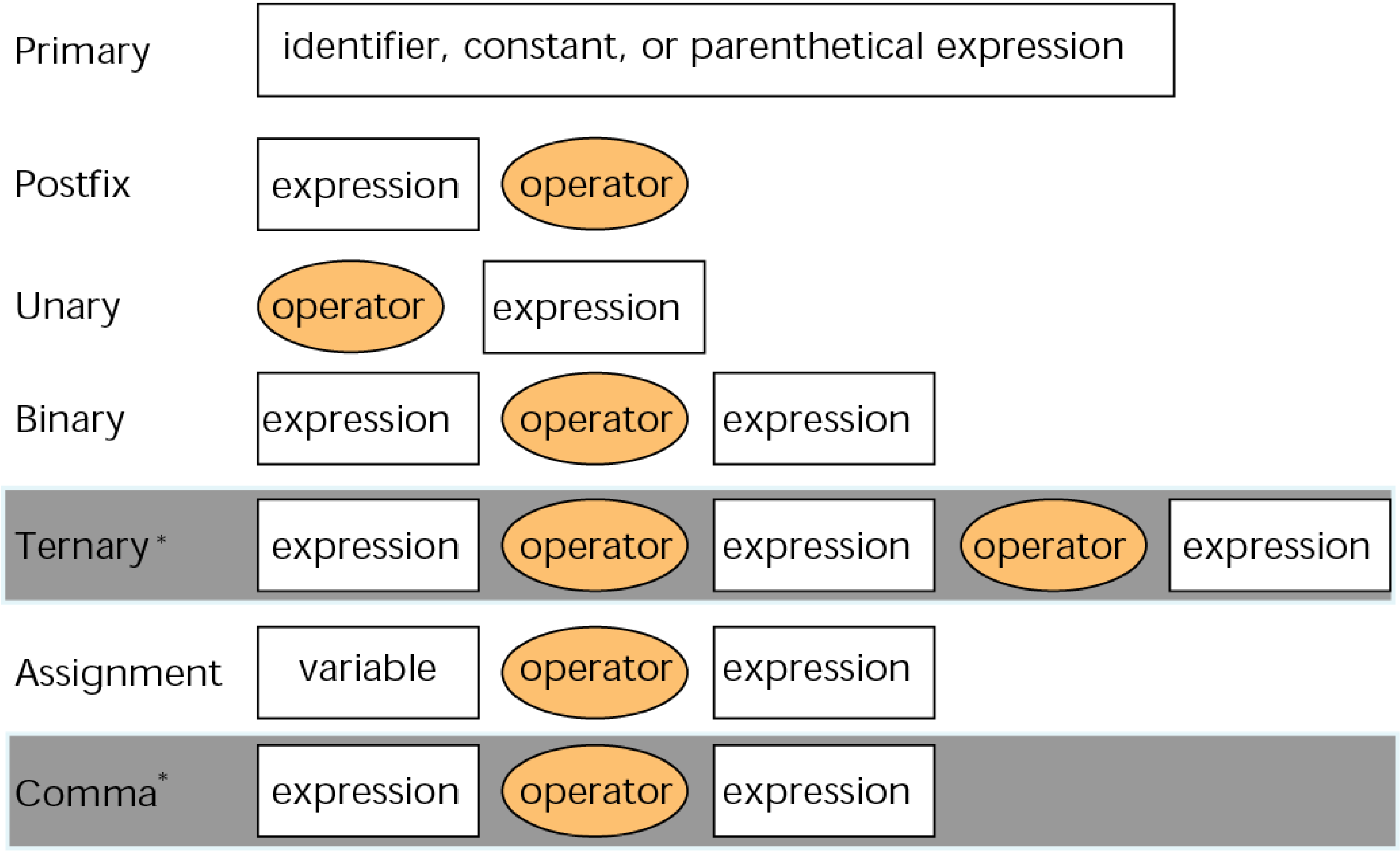
3.1

EXPRESSIONS

Note:

Expressions always reduce to a single value.

Figure 3-1 C++ expression format



Primary: identifier, constant, or parenthetical expression

Postfix: expression operator

Unary: operator expression

Binary: expression operator expression

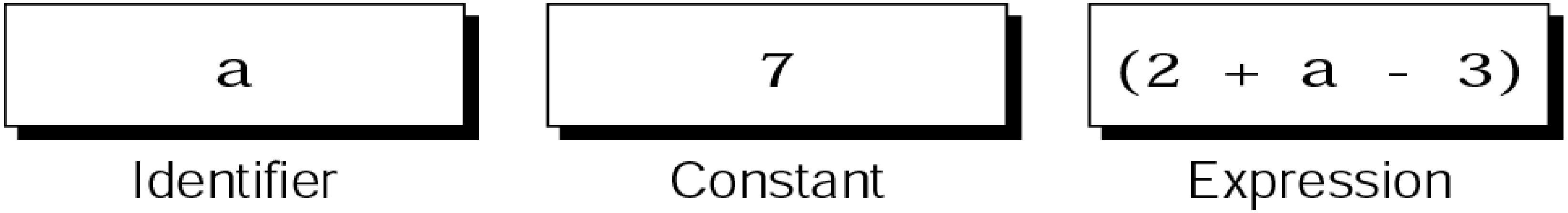
Ternary*: expression operator expression operator expression

Assignment: variable operator expression

Comma*: expression operator expression

*These expression types are unique to C and C++

Figure 3-2 Primary expressions



a

Identifier

7

Constant

(2 + a - 3)

Expression

Brooks/Cole
Thomson Learning™

Figure 3-3 Binary expressions

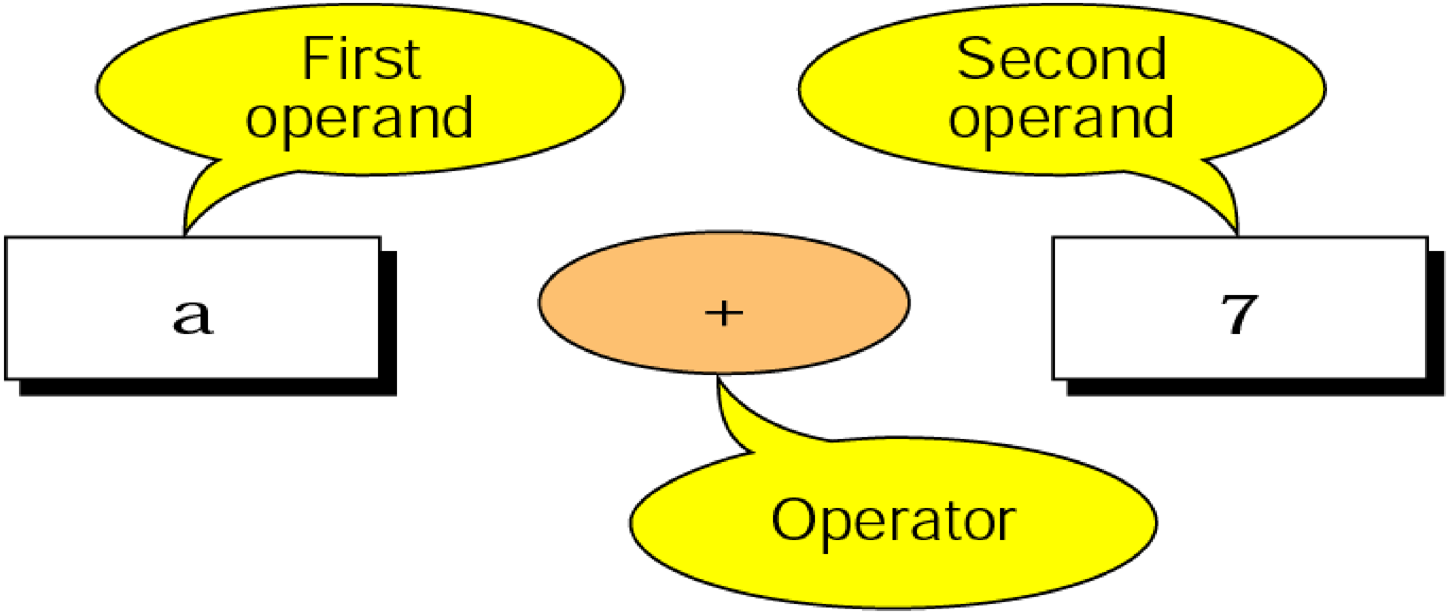
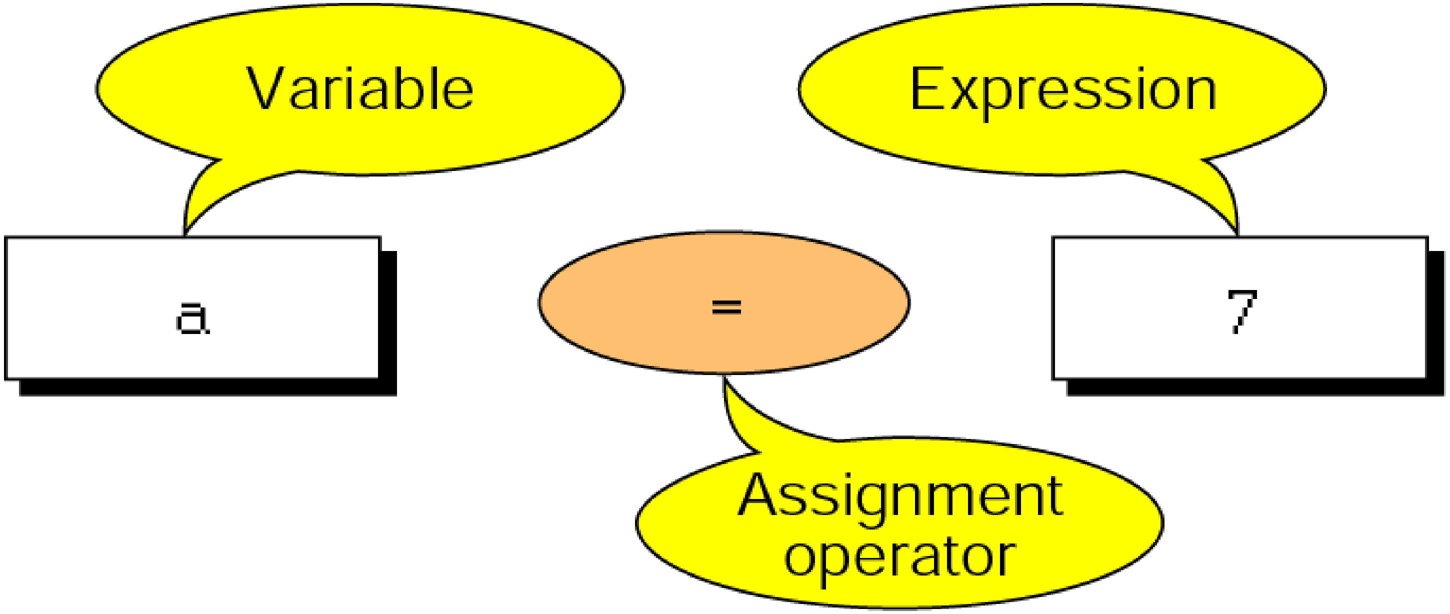


Figure 3-4 Assignment expression



Note:

The left operand in an assignment expression must be a single variable.

Note:

Assignment Expression

The assignment expression has a value and a result.

- ***The value of the total expression is the value of the expression on the right of the assignment operator (=).***

- ***The result places the expression value in the operator on the left of the assignment operator.***

Figure 3-5 Postfix expressions

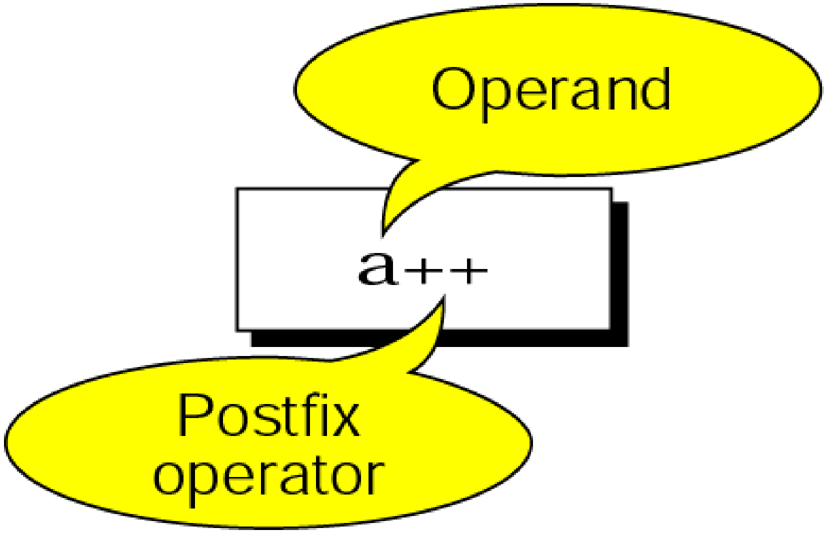


Figure 3-6 Result of postfix a++

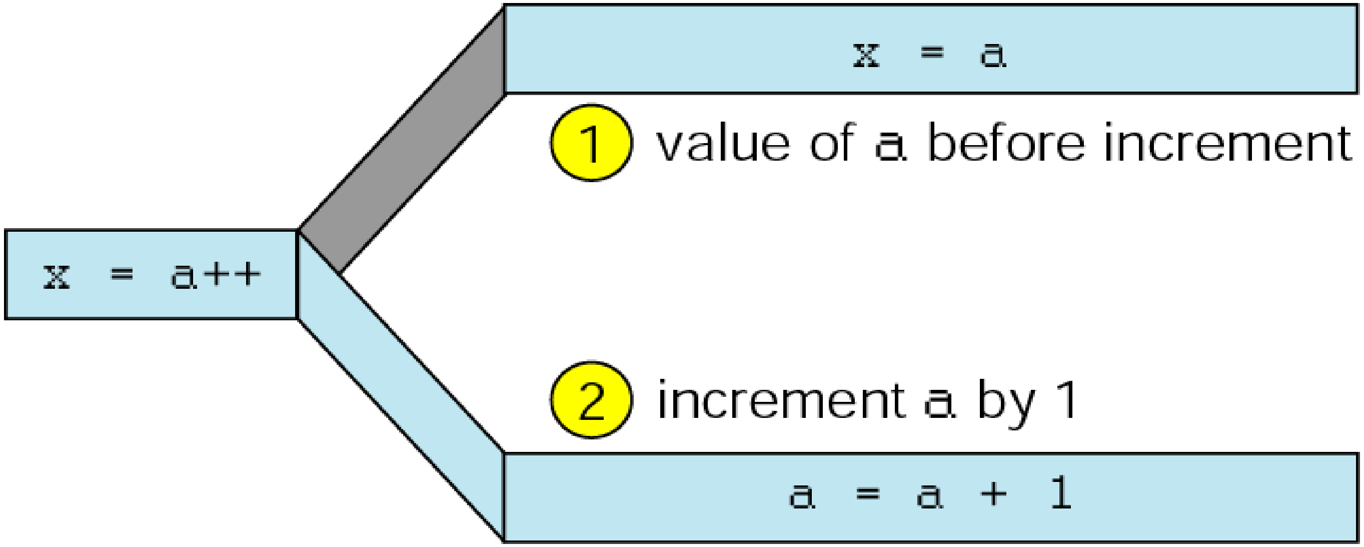


Figure 3-7 Unary expressions

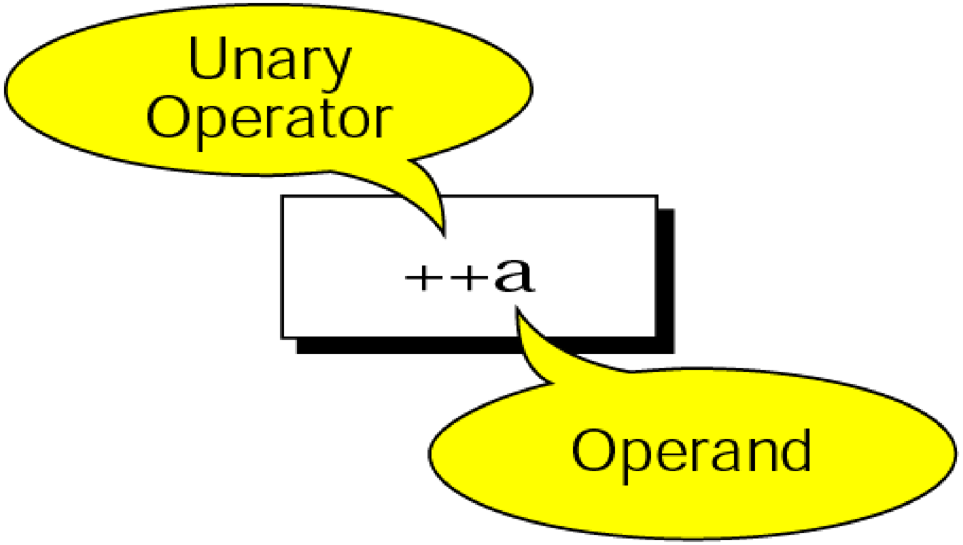
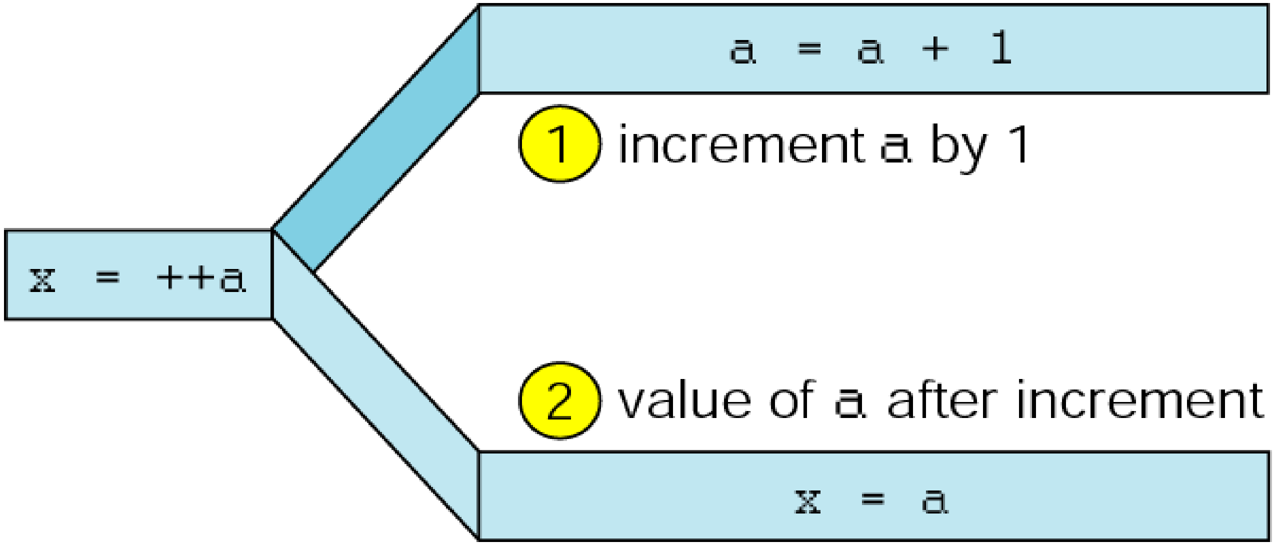


Figure 3-8 Result of prefix ++a

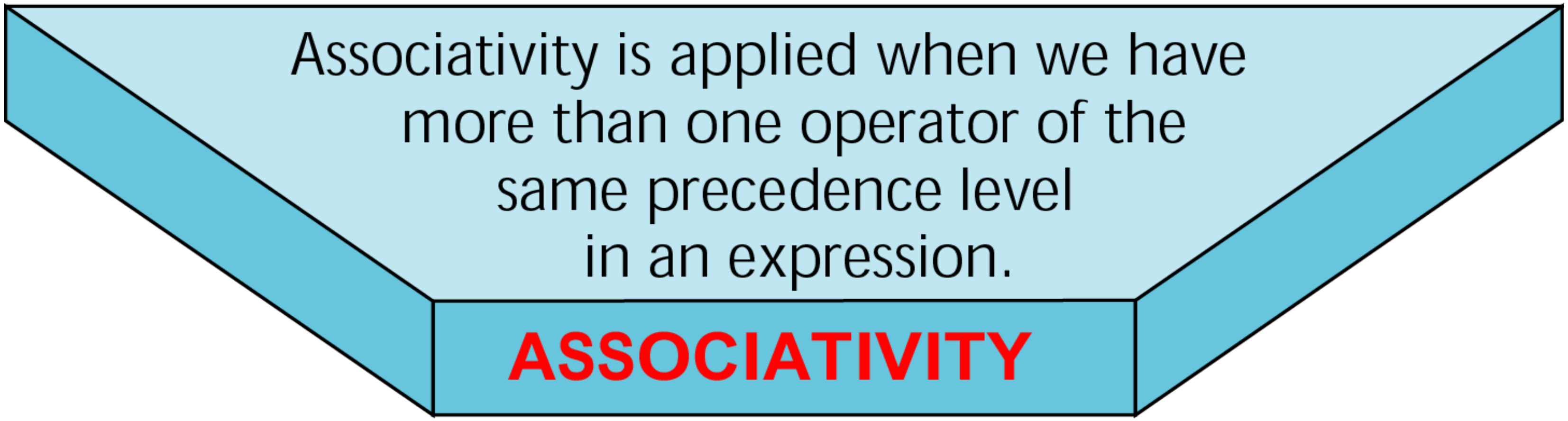


Note:

(++a) has the same effect as
(a = a + 1)

Precedence and Associativity

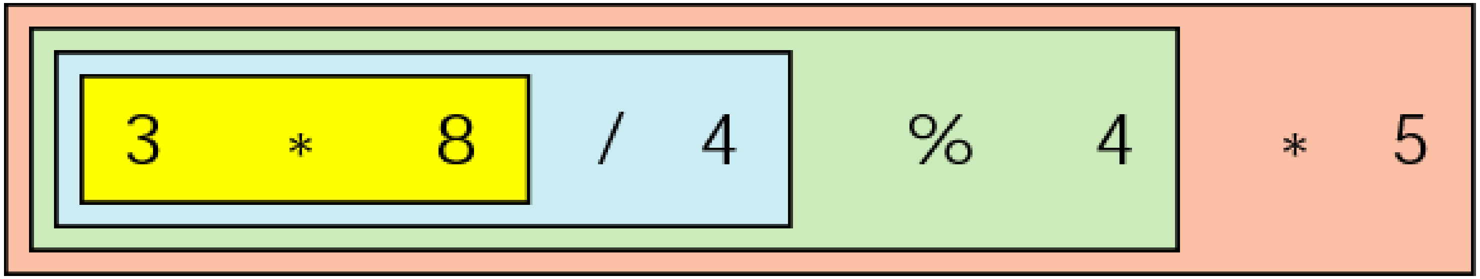
Figure 3-9 Associativity



Associativity is applied when we have more than one operator of the same precedence level in an expression.

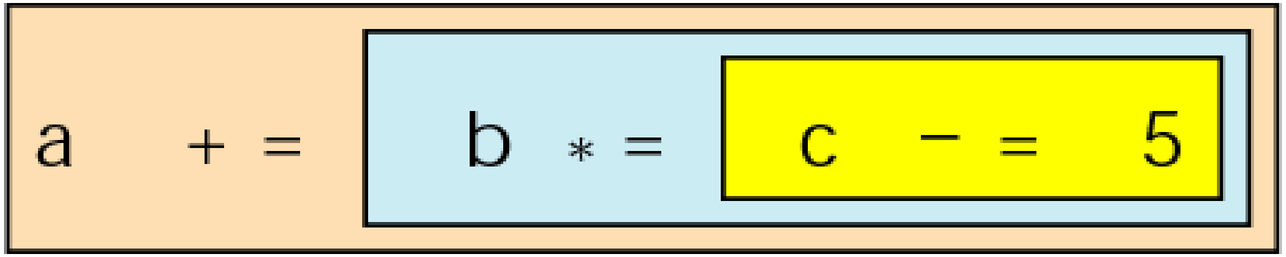
ASSOCIATIVITY

Figure 3-10 Left associativity



Brooks/Cole
Thomson Learning™

Figure 3-11 Right associativity



a += b *= c -= 5

3.3

SIDE EFFECTS

Brooks/Cole
Thomson Learning™

3.4

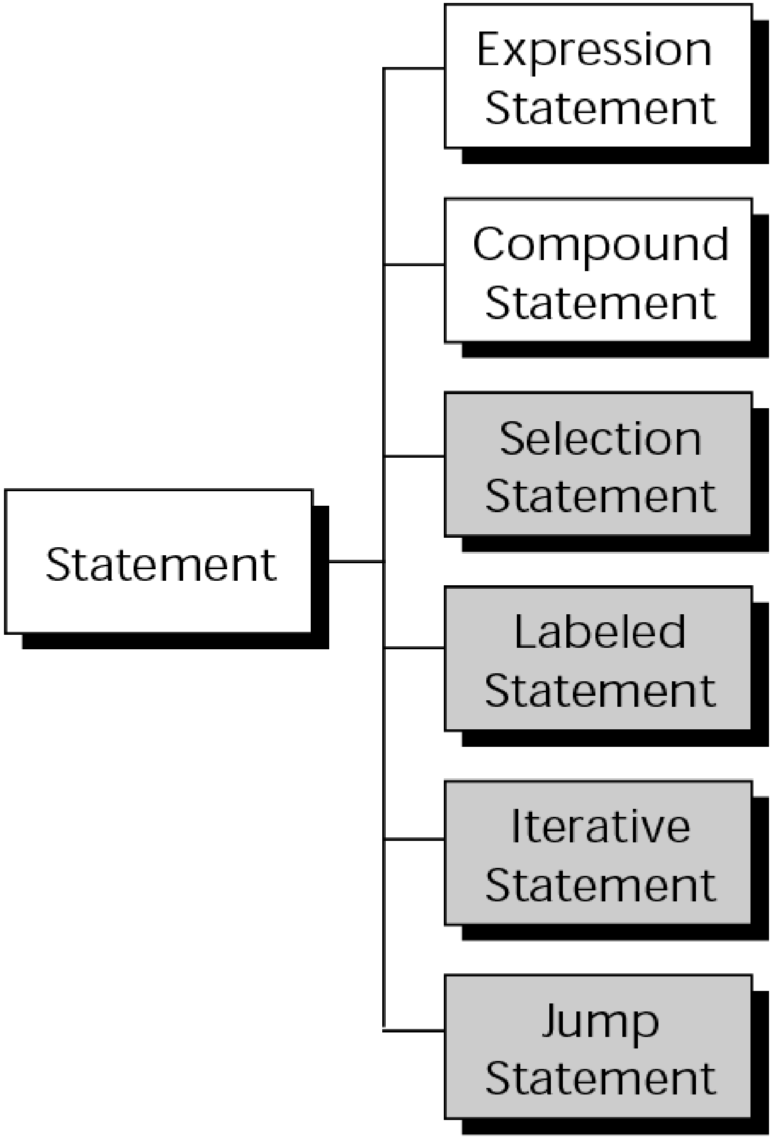
Evaluating Expressions

MIXED TYPE EXPRESSIONS

3.6

STATEMENTS

Figure 3-12 Types of statements

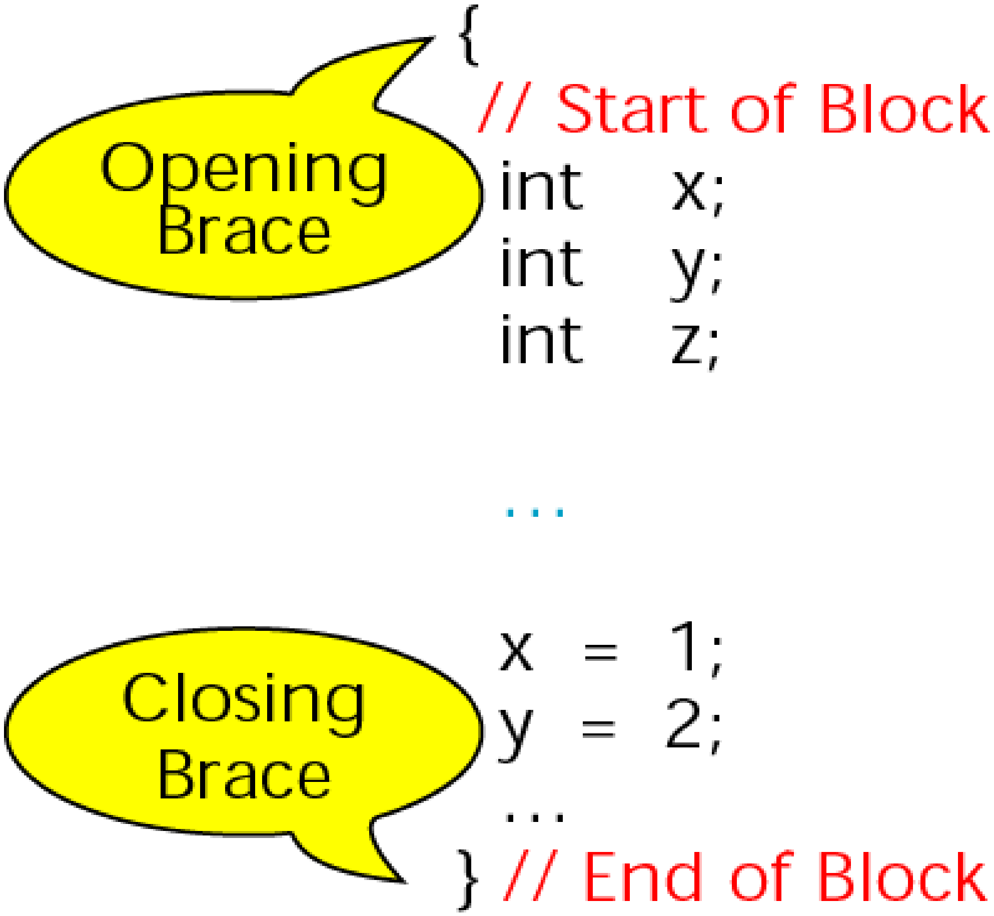


Brooks/Cole
Thomson Learning™

Note:

An expression statement is terminated with a semicolon. The semicolon is a terminator, and it tells the compiler that the statement is finished.

Figure 3-13 Compound statement



```
{
// Start of Block
int    x;
int    y;
int    z;

...

x  =  1;
y  =  2;
...
} // End of Block
```

SAMPLE PROGRAMS

3.8

Software Engineering and Programming Style

Brooks/Cole
Thomson Learning™

Note:

Blocks of code should be no longer than one screen.