C++ Expressions

Lecture 5
Sections 2.6, 2.13, 2.15

Robb T. Koether

Hampden-Sydney College

Wed, Sep 4, 2019

Outline

Object Names

- To store a quantity, it must first be given a name, called its identifier.
- Naming rules.
  - An identifier must begin with a letter.
  - The identifier may otherwise contain any combination of letters, digits, and underscores (no embedded blanks).
  - Names are case-sensitive: `cost` and `Cost` are not the same.

Object-Naming Conventions

- Naming conventions.
  - The name should be meaningful.
  - The name should begin with a lowercase letter.
  - In a multi-word name
    - Begin each subsequent word with an uppercase letter (camel style), e.g., `numberOfStudents`.
    - Or, separate the words with underscores and use all lowercase, e.g., `number_of_students`.
    - But do not write `numberofstudents`.
  - Make your program as readable and as understandable as possible.

Object Declarations

- To declare an object is to introduce its name and to specify its data type.
- An object must be declared either before or at its point of first use.
- It must be declared *once, but only once*.

Object Declarations

Object Declaration

```
int a;        // Reserves 4 unitialized bytes
float f;      // Reserves 4 unitialized bytes
char c;       // Reserves 1 unitialized byte
string s;     // Reserves 12 bytes (empty string)
```

- To write a declaration, begin the statement with the object type, followed by the name of the object.
- The compiler will reserve the appropriate number of bytes of memory for the object's value and correctly interpret that value throughout the program.

Object Initialization

Object Initialization

```
int a;                  // Uninitialized
int b = 5;              // Initialized to 5
int c = b;              // Initialized to 5
int d = b + 2*c;        // Initialized to 15
string s;               // Initialized to ""
string t = "hello" // Initialized to "hello"
```

- Fundamental-type objects are not automatically initialized.
- `string` objects are automatically initialized to the empty string.
- An object may be initialized using either a literal, or another named object, or an expression.

Outline

Unary Operators

- A unary operator operates on a single object.
- Some unary operators for **`int`**s and **`float`**s are
  - Negation: $-$
  - Increment: $++$
  - Decrement: $--$

Binary Operators

- A binary operator operates on a pair of objects called the left operand and the right operand.
- Some binary operators for **`int`**s and **`float`**s are
  - Addition: $+$
  - Subtraction: $-$
  - Multiplication: $*$
  - Division: $/$

The Remainder Operator

- The remainder operator `%`, also called the mod operator, gives the remainder when the left operand is divided by the right operand.
- The remainder operator may be applied only to integer types.
- Numerical examples
  - `20 % 3 = 2`
  - `50 % 5 = 0`
  - `3 % 10 = 3`

Outline

Expressions

- An expression consists of any legitimate combination of objects and operators.
- For example,

```
a + 2*b
```

- Every expression(and subexpression) has a value and a type.

Expression Types

- If an arithmetic expression consists only of integers, then the result is an integer.
- If an arithmetic expression consists only of floating-point numbers, then the result is a floating-point number.
- The one tricky case is integer division:

  An integer divided by an integer is an integer.

Outline

Precedence

- When two *different* operators are used in an expression, a precedence rule tells which one is done first.
- Among negation, remainder, `+`, `-`, `*`, and `/`,
  - negation `-` has the highest precedence.
  - `*`, `/`, and remainder have next-lower precedence.
  - `+*` and `-` have lowest precedence.
- In the absence of parentheses, the operator with higher precedence is done before the operator with lower precedence.

Associativity

- When the *same* operator is used twice in an expression, an associativity rule tells which one is done first.
- Left associativity – perform operations from left to right.
- Right associativity – perform operations from right to left.
- Negation is right associative.
- All the others are left associative.

Summary of Precedence and Associativity Rules

- Addition and subtraction have the same precedence.
- Multiplication, division, and remainder have the same precedence, which is higher than addition and subtraction.
- These five operators are all left associative.
- Unary operators have the highest precedence and they are right associative.
- In all other cases, use parentheses to specify the order or look it up.

Outline

Sample Programs

- Example
  - `QuadraticRoots.cpp`

Outline

Assignment

Assignment

- Read Sections 2.6, 2.13, 2.15