Friends and Unary Operators

Lecture 11
Sec 4.5

Robb T. Koether

Hampden-Sydney College

Wed, Feb 11, 2009
Definition (Friend)

A **friend** of a class is a function or a class that is given access to the private members of that class through the `friend` keyword.

- The class must declare who its friends are.
Operators as Friends

- To make an operator a friend,
  - Write
    ```
    friend Function(Parameters);
    ```
    or
    ```
    friend Class-name;
    ```
  in the class definition.
Writing Operators as Friends

- Declare the operator to be a friend of the class.
- Write the operator as a non-member function.
- Then the operator may access the data members of the operands directly.
Operators as Friends: Considerations

- **Advantages**
  - Only one function call is needed.
  - The operator has direct access to the data members.

- **Disadvantages**
  - Friendship violates the data-hiding principle.
Choosing a Method

- The preferred method is to use facilitators.
- Exceptions
  - Operators that must be member functions.
  - Unary operators.
  - Binary operators in which the left operand will always be an object of the class.
- In the exceptional cases, write the operator as a member function.
- Only in very rare cases will we use friends.
Unary Operators

- Unary operators should be implemented as member functions.
- The operator is invoked by a single operand.
- The expression \(*a\) is interpreted as \(a\).operator*()
- There is no issue of left operand vs. right operand.
The Pre-Increment Operator

The pre-increment operator should return the object *by reference*.

The expression uses the returned value.

What will `++(;++a)` do?
The Post-Increment Operator

- The post-increment operator should return the object by value.
- Include one unused and unnamed int parameter to distinguish post-increment from pre-increment.
- The designers of C++ apologize for this completely artificial mechanism.
The Post-Increment Operator

```
Type Type::operator++(int)
{
    Type original = *this;
    // Increment the object
    :
    return original;
}
```

- The expression uses the returned value.
- What will `(a++)++` do?
- What about `++(a++)` and `(++a)++`?
Example (IncrementTest.cpp)

- Download and run IncrementTest.cpp.
- It will test \( ++(++a), (++a)++, ++(a++), (a++)++ \), etc.
Assignment

Homework

Read Section 4.5, pages 169 - 180.