Assignment Operator Overview Solutions

The Assignment Operator

- Explain what an assignment operator is
  - An assignment operator sets the members of an existing object to have the same values as another object of the same class
- What is the prototype of the assignment operator?

```
T& operator =(const T& other);    // Assignment operator for type T
```

- How is it invoked?
  - Whenever we write a statement such as

    ```
    a = b;
    ```
  - The compiler will generate code which calls the operator with the appropriate argument
  - The operator is a member function, so it will be called as

    ```
    a.operator=(b);
    ```

Multiple Assignment

- How is a statement such as

```
x = y = z;
```

- processed?
  - The statement is processed from right to left (the opposite of most operators)

    ```
    x = (y = z);
    x = (y.operator=(z));
    x.operator(y.operator=(z));
    ```

  - After the statement is executed, x will have the same value as y, which has the same value as z

Return Value of Assignment Operator

- Why does the assignment operator return the modified value?
  - So that assignment operators can be chained

```
x = y = z;          // Does not work if y returns original value
```

- Why is this value not returned as a const reference?
  - To be consistent with other operators in C++ which return modifiable references
  - The class cannot be used with standard library containers if the return type is const reference

When to Write an Assignment Operator

- Explain why it is not normally necessary to implement an assignment operator when writing a class
  - If we do not provide an assignment operator, the compiler will generate a default assignment operator which
  - Assigns data members which are built-in types
  - Calls the assignment operator for members which are classes
- In what circumstances is it necessary?
  - When the default is not good enough
  - Usually this is when the class manages a resource