

Overview of Assignment Operator Solutions

- 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);`

- How is a statement such as

`x = y = z;`

- processed?
 - It 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

- 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?
 - Consistent with other operators in C++ which return modifiable values
 - Cannot be used in standard library containers if the return type is non-const reference

- 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 of 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

- What factors should be considered when implementing an assignment operator?
 - If there are any members which cannot safely be overwritten, we must delete them
 - We must obtain a new instance of these members, then initialize them from the corresponding members in the object we are being assigned to
 - We must check for self-assignment before deleting anything