Operator Overloading

- Built-in operators are implemented for fundamental types or pointer types
- Many of these operators make sense for other types of data, such as vectors or matrices, but in C, they must be implemented using ordinary functions
- Most operators in C++ can be overloaded to simplify these operations
- Overloads are functions with operands as arguments that return the result
Rules for Operands

• Overloads of [] and = must have left operand of class type
• If left operand is of class type, overload should be in class definition, and left operand is current object, not included in argument list
• If left operand not of class type, or of class type that can't be modified, overload is declared outside of any class, with all operands in argument list

Conversion Operators

• Overloading assignment operators is useful for implementing conversions
• Example:
  Complex& operator=(double x);
• Can have unintended consequences if multiple conversions can apply
• Can resolve ambiguity using explicit casts, adding conversion functions, or even more overloading to provide better match for compiler
Function Objects

- The () operator, used for calling functions, can also be overloaded
- This allows an object to be treated as a function
- Resulting function objects can store data in private members, allowing for persistence between calls and shorter argument lists
- Like [] and =, can only be overloaded for objects of class type

Next Time

All about sequential containers
- vectors, lists, stacks and queues
- More about iterators
- More operations on strings