# OBJECT ORIENTED PROGRAMMING

Lab 3

# OBJECT ORIENTED PROGRAMMING WITH C++

- Chapter Objectives
  - Assigning objects
  - Passing objects to functions
  - Returning objects from functions
  - An Introduction to friend functions

# OBJECT ORIENTED PROGRAMMING WITH C++

- Assigning Objects
  - One object can be assigned to another object that both objects are of the same class type.
  - By default, when one object is assigned to another, a bitwise copy of all the data members is made.

- Passing objects to functions
  - Objects can be passed to functions as arguments in just the same way that other types of data are passed.
  - Simply declare the function's parameter as a class type and then use an object of that class as an argument when calling the function.
  - As with other types of data, by default all objects are passed by value to a function.

- Passing objects to functions
  - When a copy of an object is made when being passed to a function, it means that a new object comes into existence.
  - When a copy of an object is made to be used in a function call, the constructor function is not called.
  - Also, when the function that the object was passed to terminates, the copy of the argument is destroyed, and the destructor function is called.

- Returning objects from functions
  - Functions can return objects by declaring object's class type for the function's returning type.
  - And return an object of that class type using the normal **return** statement.
  - The key point is that when an object is returned from a function, the temporary object used to effect the return will have its destructor function called.
  - Thus, you should avoid returning objects in which this situation is harmful.

# OBJECT ORIENTED PROGRAMMING WITH C++

- An Introduction to friend functions
  - A friend is not a member of a class but still has access to its private elements.
  - Two reasons that friend functions are useful have to do with operator overloading and the creation of certain types of I/O functions.
  - A third reason for friend functions is that there will be times when you want one function to have access to the private members of two or more different classes.