Classes and Objects

Lecture 34
COP 3014 Spring 2017

April 11, 2017

Object Oriented Languages

- A program is a list of instructions for a computer to execute.
- Object-oriented languages, like C++ and Java, take this a step further and encapsulate their data and procedures together in units called objects, which contain more than just functions (actions, often representable by verbs).
- These languages make items modular as well (objects, or things representable by nouns).
- Object-oriented languages are also high-level languages, more readable for people, and needing translation for the machine (by a compiler or interpreter).

Classes and Objects:

- **Object** – an encapsulation of data and functions that act upon that data. The three aspects of an object:
  - *Name* – the variable name we give it
  - *Attributes* (member data) – the data that describes the what the object is
  - *Behavior* (member functions) – behavior aspects of the object (functions that describe what the object does)
- **Class** – a blueprint for objects. A class is a user-defined type that describes what a certain type of object will look like. A class description consists of a declaration and a definition (usually split into separate files).
- An object is a single instance of a class.
- You can create many objects from the same class type, in much the same way that you can create many houses from the same blueprint.

Classes and Objects: Another Definition

- **Object** – an encapsulation of data along with functions that act upon that data.
- An object consists of:
  - Name – the variable name we give it
  - Member data – the data that describes the object
  - Member functions – behavior aspects of the object (functions related to the object itself)
- **Class** – a blueprint for objects. A class is a user-defined type that describes what a certain type of object will look like. A class description consists of a declaration and a definition. Usually these pieces are split into separate files.
- An object is a single instance of a class. You can create many objects from the same class type.

DDU Design - Declare, Define, Use:

- **Declare**
  - A declaration gives an interface. A variable declaration gives the type. A function declaration tells how to use it, without bothering with how it works.
  - A class declaration shows what an object will look like and what its available functions are. Again, implementation details aren't needed here.
- **Define**
  - A definition usually consists of the implementation details. This part doesn't need to be seen by the user of the interface.
  - A function definition is the code that makes a function work (the function body).
  - A class definition consists definitions of its members.

DDU Design - Declare, Define, Use:

- **Use**
  - The use of an item, through its interface. The user of an executable program uses the graphic interface, keyboard, and mouse.
  - The user of a function is a programmer, who makes calls to the function (without needing to know the implementation details).
  - The user of a class is also a programmer, who uses the class by creating objects and calling the available functions for those objects.
- **Interface**
  - The concept of interface is a very important one in object-oriented programming.
  - The interface is what the user sees. We often leave the implementation details hidden.
  - We want to strive to create a clear interface for the user (not necessarily the end user of a program – could be a programmer, i.e. the user of a class).

Protection levels in a class:

- We can declare members of a class to be public or private.
  - public - can be accessed from inside or outside of the object.
  - private - can only be used by the object itself.
- The public section of a class is essentially the interface of the object.
- The user of an object is some other portion of code (other classes, functions, main program).
- So, objects are used by programmers, and we want the interface to be as simple as possible.
- This usually means providing functions in the public area that handle all of the necessary actions on the object.
- Although there is no set rule on what is made public and what is made private, the standard practice is to protect member data of a class by making it private.
- If there are functions that do not need to be part of the interface, then these can be private, as well.

Data Hiding

Data Hiding is the concept of acccessing data on a Need-to-Know basis. If a class or a function does not need access to a certain piece if data, then it is unaware of its existence.

Reasons for data hiding:

- Makes interface simpler for user. Principle of least privelege (need-to-know)
- More secure. Less chance for misuse (accidental or malicious).
- Class implementation easy to change without affecting other modules that use it.

Class Declaration Format

```
class <className>
{
     private:
     (private member data and functions go here)
     public:
     (public member data and functions go here)
};
```

Please look at today's example for the Point Class

Constructors:

- A constructor is a special member function of a class whose purpose is usually to initialize the members of an object.
- A constructor is easy to recognize because:
  - It has the same name as the class
  - It has no return type
- A constructor is a function, and you can define it to do anything you want.
- However, you do not explicitly call the constructor function. It is automatically called when you declare an object.
- The term default constructor will always refer to a constructor with no parameters.
- When we declared objects in example, its default constructor was used because no parameters were passed.
- To utilize a constructor with parameters, we can pass arguments in when the object is declared.

Accessors and Mutators

- **Accessors**: These are public member functions that take no parameters and return one member data.
- By convention, they begin with "get" followed by the name of the member data.
- **Mutators**: These are public member functions that can be used to change one data member.
- They accept one parameter of the same type as the member data, and return nothing.
- By convention, they begin with "set" followed by the name of the member data.