Intoduction to C++

Programming in C++

- C++
  - Improves on many of C's features
  - Has object-oriented capabilities
    - Increases software quality and reusability
  - Developed by Bjarne Stroustrup at Bell Labs
    - Called "C with classes"
    - C++ (increment operator) - enhanced version of C
  - Superset of C
    - Can use a C++ compiler to compile C programs
    - Gradually evolve the C programs to C++

Clean Interface

- The emphasis is on creating a set of tools which can be used cleanly, with a minimum knowledge about implementation in the user's driver files. The following concepts are relevant to accomplishing clean interface:

- **Data Abstraction**
  - Define an object by its data and allowable operations: an abstract data type.
- **Information hiding**
  - Restrict access to data so that it can be manipulated only in authorized ways. Separate class declarations from implementation.
- **Encapsulation**
  - Bundle data and operations into one logical unit.

C++ Techniques

- ➢ Relevant techniques include:

1. C++ <u>classes</u>, with *private* and *public* <u>members</u>
2. Function and operator name <u>overloading</u> to give "natural" function calls
3. <u>Templates</u> to allow the same code to be used on a variety of different data types
4. A clean <u>built-in I/O interface</u>, which itself involves overloading the input and output operators

- ➢ Learning these techniques is much of what C++ is all about.

A Basic C++ Program

```
#include <iostream>
#include <math.h>

int main()
{
   float x;

   std::cout << "Enter a real number: " << std::endl;
   std::cin >> x;

   std::cout << "The square root of " << x << " is: "
              << sqrt(x) << std::endl;
   return 0;
}
```

Classes and Objects

- **Class:** a type definition that includes both
  - data properties, and
  - operations permitted on that data
- **Object:** a variable that
  - is declared to be of some Class
  - therefore includes both data and operations for that data
- **Appropriate usage:**

  "A variable is an instance of a type."

  "An object is an instance of a class."

Basic Class Syntax

- A class in C++ consists of its **members**.
  - A member can be either <u>data</u> or <u>functions</u>.
- The functions are called **member functions** (or **methods**)
- Each instance of a class is an **object**.
  - Each object contains the data components specified in class.
  - Methods are used to act on an object.

Class syntax - Example

```
// A class for simulating an integer memory cell

class  IntCell
{
  public:
      IntCell( )
      { storedValue = 0; }

      IntCell(int initialValue )
      { storedValue = initialValue;}

      int read( )
      { return storedValue; }

      void write( int x )
      { storedValue = x;}

  private:
      int storedValue;
};
```

constructors

Class Members

- `Public` member is visible to all routines and may be accessed by any method in any class.
- `Private` member is not visible to non-class routines and may be accessed only by methods in its class.
- Typically,
  - Data members are declared private
  - Methods are made public.
- Restricting access is known as *information hiding*.

Constructors

- A constructor is a method that executes when an object of a class is declared and sets the initial state of the new object.
- A constructor
  - has the same name with the class,
  - No return type
  - has zero or more parameters (the constructor without an argument is the *default constructor*)
- There may be more than one constructor defined for a class.
- If no constructor is explicitly defined, one that initializes the data members using language defaults is automatically generated.

Extra Constructor Syntax

```
// A class for simulating an integer memory cell

class  IntCell
{
   public:
       IntCell( int initialValue = 0 )
          : storedValue( initialValue) { }

       int read( ) const
          { return storedValue; }

       void write( int x )
          { storedValue = x; }
   private:
       int storedValue;
};
```

Single constructor (instead of two) ?

Accessor and Modifier Functions

- A method that examines but does not change the state of its object is an <u>accessor</u>.
  - Accessor function headings end with the word `const`
- A member function that changes the state of an object is a <u>mutator</u>.