Monday, February 3, 2003

MAT 102, Data Structures

Using Classes in C++
The C++ Class

• The class type, like the struct type, allows one to group together several different types.

• In addition, the class type is often used to group data and functions together.
class Date {
public:
  void setDate(int newDay, int newMonth, int newYear);
  void outputDate();
  int getDay();
  int getMonth();
  int getYear();
private:
  int day;
  int month;
  int year;
};
• Make member variables private and then provide public member functions to access the variables.

• Class need not change his/her code. This way, you can change how you store the data at some later date and the user of your class need not change his/her code.
Constructors

There can be special member functions in a class called "constructors." A constructor is called whenever a variable of the class is declared. Constructors are used usually to initialize the variables in the class. We can "overload" and create several different constructors for the same class.

Constructors can be special member functions in a class called "constructors."
// example constructor for Date

class Date {
public:
    Date(); // constructor has no return type

private:
    int day;
    int month;
    int year;

    // other member functions

    Date(); // constructor is named the same as the class

    // example constructor for Date

class Sphere, two constructors, seven other member functions, one member variable

// see sample code Sphere.h and Sphere.cpp for more info

double theRadius;
private:
    double theRadius;

void displayStatistics() const;
    double getVolume() const;
    double getArea() const;
    double getDiameter() const;
    double getTCircumference() const;
    double getTPerimeter() const;
    double getTRadius() const;

void setRadius(double newRadius);

Sphere(double initialRadius);

public:
    class Sphere

    functions, one member variable

    class Sphere, two constructors, seven other member
As you begin to create classes, you can organize the class definition in a header file and then compile the implementation file. Simply add the line `#include "header.h"` to include the class definition and then compile the header file. Then you can use the class in any C++ file by including the header file and linking to the implementation file. By putting the class definition in a header file and the member function details in an implementation file, you can organize the class files. Using header files...
5. Run your program with `./Program`.

4. Link the two *.o files together to form an executable.

3. Create Program.o from Your Program.cpp using:

   ```
   g++ -c Program.cpp
   g++ -c Sphere.cpp
   g++ Sphere.o Program.o -o Program
   ```

2. Create Sphere.cpp using the command:

1. Place `#include "Sphere.h"` in your Program.cpp.

   // do the following:
   // to use the Sphere class in another program, you
   // and an implementation file called Sphere.cpp
   // suppose you have a header file called Sphere.h
g++ -c useSphere.cpp
useSphere.o: useSphere.cpp
  g++ -c useSphere.cpp
  useSphere.o: Sphere.cpp
  Sphere.o: Sphere.cpp
  useSphere: Sphere.o useSphere.o
  useSphere.o: Sphere Sphere.o
  g++ Sphere.cpp
  useSphere: Sphere Sphere.o
  useSphere: useSphere useSphere.cpp
  // MAKEFILE to use Sphere.cpp in program //
Homework (Exercise 6)

• Copy sample code to your account on albert.
• Create a program that uses the Sphere class and computes the area of spheres of radius 1.0, 2.0, 3.0, and 4.0.
• Begin reading Chapter 3 of Carrano and Prichard.

Homework (Exercise 6)