Object-Oriented Programming Using C++

Data Hiding

Indranil Saha

Department of Computer Science and Engineering
Indian Institute of Technology Kanpur

Data Hiding

- One of the most important features of Object Oriented Programming
- Allows preventing the functions of a program to access directly the internal representation of a class type
- The access restriction to the class members is specified by the labeled public, private, and protected sections within the class body
- The keywords public, private, and protected are called access specifiers

Access Restrictions for Class Members

- **Public:**
  - A public member is accessible from anywhere outside the class but within a program
  - You can set and get the value of public variables without any member function
- **Private:**
  - Cannot be accessed, or even viewed from outside the class.
  - Only the class and friend functions can access private members.
  - By default all the members of a class would be private
- **Protected:**
  - Very similar to a private member
  - They can be accessed in child/derived classes

Example: Public Members

Public member in the Box class

```
#include <iostream>
using namespace std;

class Line {
   public:
      double length;
      void setLength( double len );
      double getLength( void );
};

double Line::getLength(void) {
   return length ;
}

void Line::setLength( double len) {
   length = len;
}

int main() {
   Line line;

   // set line length
   line.setLength(6.0);
   cout << "Length of line : " << line.getLength() <<endl;

   // set line length without member function
   line.length = 10.0; // OK: because length is public
   cout << "Length of line : " << line.length <<endl;

   return 0;
}
```

Example: Private Members

Private member in the Box class

```
#include <iostream>
using namespace std;

class Box {
   public:
      double length;
      void setWidth( double wid );
      double getWidth( void );

   private:
      double width;
};

double Box::getWidth(void) {
   return width ;
}

void Box::setWidth( double wid ) {
   width = wid;
}

....
}
```

Example: Private Members (Contd.)

Private member in the Box class

```
....

// Main function for the program
int main() {
   Box box;

   // set box length without member function
   box.length = 10.0; // OK: because length is public
   cout << "Length of box : " << box.length <<endl;

   // set box width without member function
   // box.width = 10.0; // Error: because width is private
   box.setWidth(10.0);  // Use member function to set it.
   cout << "Width of box : " << box.getWidth() <<endl;

   return 0;
}
```

Example: Protected Members

Protected member in the Box class

```
#include <iostream>
using namespace std;

class Box {
   protected:
      double width;
};

class SmallBox:Box { // SmallBox is the derived class.
   public:
      void setSmallWidth( double wid );
      double getSmallWidth( void );
};

double SmallBox::getSmallWidth(void) {
   return width ;
}

void SmallBox::setSmallWidth( double wid ) {
   width = wid;
}

int main() {
   SmallBox box;

   // set box width using member function
   box.setSmallWidth(5.0);
   cout << "Width of box : "<< box.getSmallWidth() << endl;

   return 0;
}
```

Object-Oriented Programming Using C++

Data Hiding

Indranil Saha

Department of Computer Science and Engineering
Indian Institute of Technology Kanpur