Data Structures and Object Oriented Programming using C++

Ahsan Ijaz

Inheritance

In object-oriented language, a class can obtain the properties of another class through the mechanism of inheritance.

- In C++, inheritance is supported by class derivation.
- A derived class inherits the properties of its base class.

Inheritance

In object-oriented language, a class can obtain the properties of another class through the mechanism of inheritance.

- In C++, inheritance is supported by class derivation.
- A derived class inherits the properties of its base class.

Vehicle

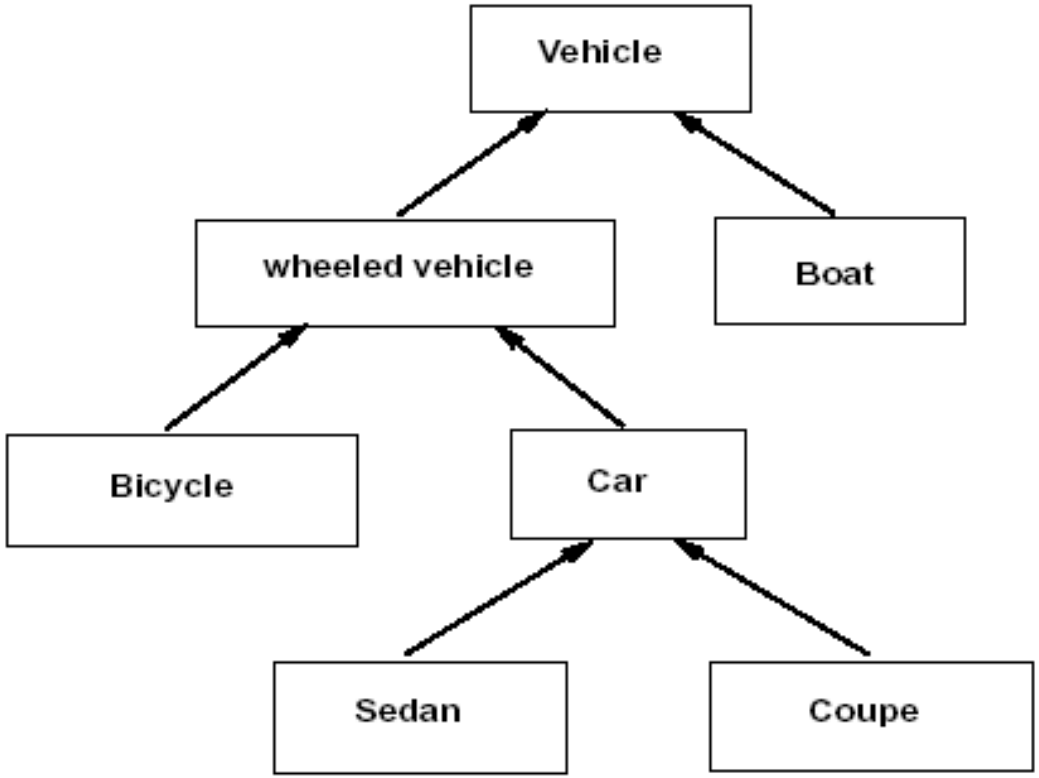

Figure: Inheritance Vehicle Example

Inheritance

- The class that is being inherited from is called the base class.
- The new class is called the derived class.
- The set of the base and derived class instances is called the class inheritance hierarchy.

Inheritance

A derived class inherits data members and member functions of its base class, and can also add its own list of generalization and specialization.

```
1 class derived-class: access-specifier base-class
```

Generalization and Specialization

Generalization:

- Extend the behavior of the base class.
- Add new member functions and/or data to derived class.

Specialization:

- Modify the behavior of the base class.
- Change implementations in the derived class without changing the base class interface.

Benefits of Inheritance

Increase software reuse and quality

- Programmers can reuse the base classes instead of writing new classes
- Using well-tested base classes helps reduce bugs in software.
- Reduce code size.

Enhance software extensibility and comprehensibility

- Helps support more flexible and extensible architectures.
- Useful for modeling and classifying hierarchically related domains.

Basic syntax of inheritance

```
1   class Pen
2   {public:
3   void SetLocation(int, int);
4   void SetStatus(int);
5   private:
6   int x, y, status;
7   };    //end class declaration.
8
9   class ColorPen : public Pen
10  {public:
11  void SetColor(int);
12  private:
13  int color;
14  };
```

Here Pen is the base class. ColorPen is the derived class. The keyword public indicates here the type of inheritance is public.

Pen inheritance

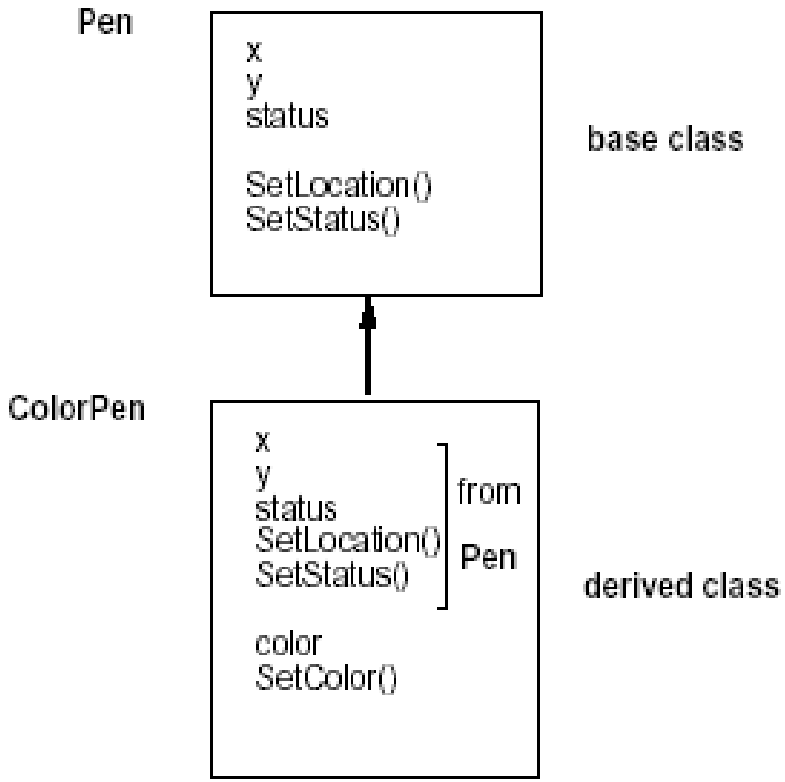

Figure: Variables of ColorPen

Derived Class composition

A derived class object consists of sub-objects of its base classes and a derived class portion.

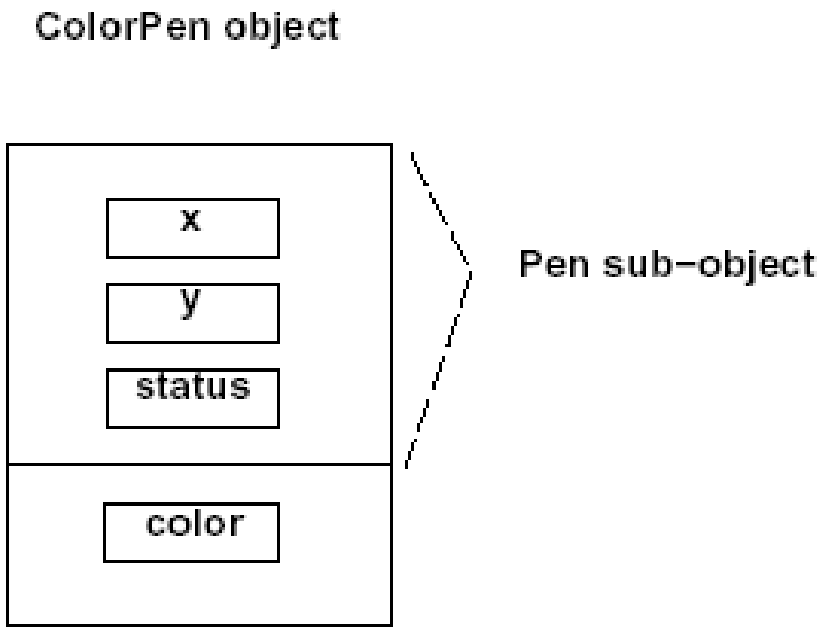

Figure: Sub-Objects

Member Access

Public Member:

- In public inheritance, public members from the base class are public in its derived classes.

Private Member:

- Private members in a base class are accessible only in the base class; they are not accessible in its derived classes.

Member access

A derived class object consists of sub-objects of its base classes and a derived class portion.

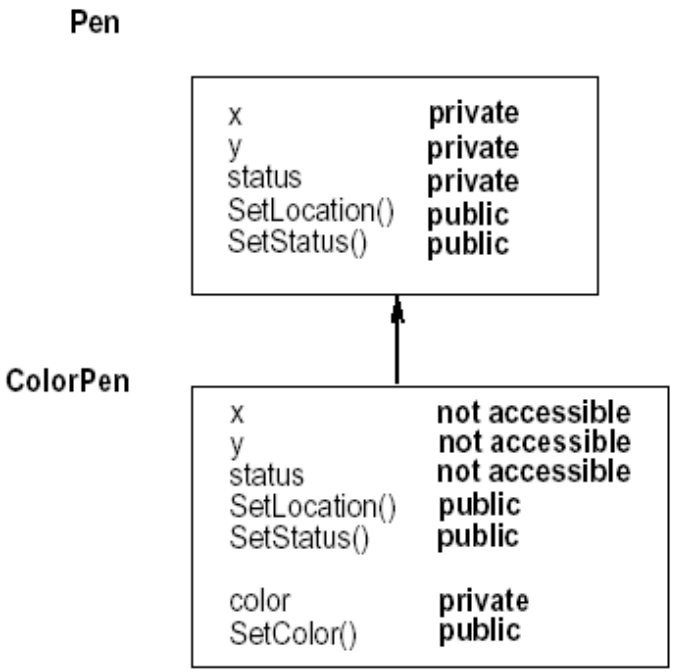

Figure: Member access in ColorPen

Member access Error

```
1  class Pen {
2  public:
3  void SetLocation(int, int);
4  void SetStatus(int);
5  private:
6  int x, y, status;};
7  class ColorPen : public Pen {
8  public:
9  void SetColor(int);
10 void setX(int xx){ x = xx; }//Error!
11          private:
12 int color;
13 };
```

Member Access

The private member x, y, status in class Pen can be accessed only by the implementers of class Pen.