Chapter 8 Inheritance, Polymorphism, and Interfaces

Polymorphism makes objects behave as you expect them to and allows you to focus on the specifications of those behaviors. Inheritance will enable you to use an existing class to define new classes, making it easier to reuse software. Finally , interfaces allow you to specify the methods that a class must implement;

Inheritance allows you to define a very general class and then later define more specialized classes that add some new details to the existing general class definition; This saves work, because the more specialized class inherits all the properties of the general class and you only need program the new features;

A student is a person , so we define the class Student to be a derived class or subclass of the class Person. A derived class is a class defined by adding instance variables and methods to an existing class. The derived class extends the base class or superclass;

Public class Student extends Person

A derived class such as Student, can also add some instance variables or methods to those it inherits from its base class .

Although Student does not inherit the private instance variable name from Person, it does inherit the method setName and all the other public methods of the base class;

Inheritance should define a natural is-a relationship between two classes;

If an is-a relationship does not exist between two proposed classes, do not use inheritance to derive one class from the other. Instead, consider defining an object of one class as an instance variable within the other class. That relationship is called has-a.

A base class is often called a parent class. A derived class is then called a child class. A child class inherits public instance variables and public methods from its parent class.

Ancestor class descendant

You define a derived class or subclass by starting with another already defined class and adding ( or changing ) methods and instance variables. The class you start with is called the base class, or superclass; the derived class inherits all of the public methods and public instance variables from the base class and can add more instance variables and methods;

Public class Derived\_Class\_Name extends Base\_Class\_Name

{

Declarations\_of\_Added\_Instance\_Variables

Definitions\_of\_Added\_And\_Changed\_Method

}

Changed methods are said to be overridden;

Overriding a method redefines it in a descendant class;

A method overrides another if both have the same name , return type ,and parameter list ;

Overriding method definitions:

In a derived class, if you include a method definition that has the same name , the exact same number and types of parameters and the same return type as a method already in the base class, this new definition replaces the old definition of the method when objects of the derived class receive a call to the method.

When overriding a method definition, you cannot change the return type of the method . Since the signature of the method does not include the return type, you can say that when one method overrides another, both methods must have the same signature and return type.

A method overloads another if both have the same name and return type but different parameter lists;