

Solutions to Quick Check Questions

14

Inheritance and Polymorphism

14.1 Defining Classes with Inheritance



Quick Check

1. Which is the subclass and which is the superclass in the following declaration?

```
class X extends Y { ... }
```

X is the subclass, and Y is the superclass.

2. Which visibility modifier allows the data members of a superclass to be accessible to the instances of subclasses?

*The purpose of the access modifier **protected** is to allow the access to the data members of a superclass from the instances of the subclasses, but disallow the access from the outside classes. The access modifier **public**, of course, will allow access to all other classes including the subclasses, but the purpose of the **public** modifier is not allowing access to the subclasses.*

14.2 Using Classes Effectively with Polymorphism



Quick Check

1. Suppose Truck and Motorcycle are subclasses of Vehicle. Which of the following declarations are invalid?

```
Truck      t = new Vehicle();  
Vehicle    v = new Truck();  
Motorcycle m1 = new Vehicle();  
Motorcycle m2 = new Truck();
```

The declarations for t, m1, and m2 are invalid.

2. What is the purpose of the instanceof operator?

To determine whether an object is an instance of a given class or not.

14.3 Inheritance and Member Accessibility



Quick Check

1. If X is a private member of the Super class, is X accessible from a subclass of Super?

No. A private member of a class is only accessible within the class.

2. If X is a protected member of the Super class, is X of one instance accessible from another instance of Super? What about from the instances of a subclass of Super?

Yes. Data members of one instance are fully accessible from any other instances of the same class.

Yes. Protected (and public) members of a superclass are accessible from the instances of a subclass.

14.4 Inheritance and Constructors



Quick Check

1. How do you call the superclass's constructor from its subclass?

By using the reserved `super` and passing the correct number and types for the arguments. Also, the call to `super` must be the first statement in the subclass constructor.

2. What statement will be added to a constructor of a subclass if it is not included in the constructor explicitly by the programmer?

The reserved word `super` with no arguments.

3. Modify the definition of `GraduateStudent` and `UndergraduateStudent` in Section 14.1 so we can create their instances in the following way:

```
student1 = new UndergraduateStudent();
student2 = new UndergraduateStudent("Mr. Espresso");
student3 = new GraduateStudent();
student4 = new GraduateStudent("Ms. Latte");
```

Add the following constructors:

```
class UndergraduateStudent extends Student
{
    public UndergraduateStudent( )
    {
        super( );
    }

    public UndergraduateStudent( String name )
    {
        super( name );
    }
    . . .
}

class GraduateStudent( )
{
    public GraduateStudent( )
    {
        super( );
    }
}
```

```

    }

    public GraduateStudent( String name )
    {
        super( name );
    }
    . . .
}

```

14.5 Abstract Superclasses and Abstract Methods



Quick Check

1. Can you create an instance of an abstract class?

No.

2. Must an abstract class include an abstract method?

No. The class could inherit abstract methods from its superclass.

3. What is wrong with the following declaration?

```

class Vehicle
{
    abstract public getVIN();
    ...
}

```

The Vehicle class must be declared as abstract because it contains an abstract member.

14.6 When and When Not to Use Inheritance

No Quick Check Questions.

14.7 Sample Program: Computing Course Grades

No Quick Check Questions.