1. Polymorphism :: Substituting a Subclass Object for a Superclass Object

Consider the *Square* and *Rectangle* classes. Since a *Square* **is a** *Rectangle* it is permissible to substitute a *Square* object where a *Rectangle* object is specified:

```
public void someMethod(Rectangle pRect) {
   ...
}

public void someOtherMethod() {
  Rectangle rect = new Rectangle(10, 20, 30, 40);
  someMethod(rect);
  Square square = new Square(10, 20, 30);
  someMethod(square);
}
```

This substitution is permissible because *Square* is a subclass of *Rectangle* which means that a *Square* object contains all of the same instance variables that a *Rectangle* object would have.

1. Polymorphism :: Substituting a Subclass Object for a Superclass Object (continued)

Note that the converse is not allowed; a *Rectangle* object cannot be substituted for a *Square* object:

```
public void someMethod(Square pSquare) {
  ...
}

public void someOtherMethod() {
  Rectangle rect = new Rectangle(10, 20, 30, 40);
  someMethod(rect);  // Syntax error. A Rectangle cannot substitute for a Square
}
```

Why is this not allowed? Consider these two classes:

```
public class Super {
  private int mX;
  // Assume accessor/mutator methods for mX are declared here.
  public void aMethod() { ... }
}

public class Sub extends Super {
  private int mY;
  // Assume accessor/mutator methods for mY are declared here.
  @Override
  public void aMethod() { ... }
}
```

1. Polymorphism :: Substituting a Subclass Object for a Superclass Object (continued)

Suppose we declare a *Super* object and a *Sub* object:

```
public void someMethod(Super pObj) {
  pObj.setX(10);  // This would be legal if pObj is actually a Sub
}

public void someOtherMethod(Sub pObj) {
  pObj.setY(20);  // This would be illegal if pObj is actually a Super
}

Super super = new Super();
Sub sub = new Sub();
someMethod(super);          // Legal: super is an object of Super
someMethod(sub);            // Legal: a Sub is a Super
someOtherMethod(sub);       // Legal: sub is an object of Sub
someOtherMethod(super);     // Illegal: a Super is not a Sub
```

In general, a subclass object may contain instance variables and instance methods that are not part of superclass objects.