Java Programming

2-2: Java Class Design – Abstract Classes

Practice Activities

Lesson Objectives:

- Use Abstract Classes
- Use the instanceof operator to compare object types
- Use virtual method invocation
- Use upward and downward casts

Vocabulary:

Identify the vocabulary word for each definition below.

downcast	The type of casting that changes a generalized object to a more specialized object type.
virtual method invocation	The process of a call to a generalized method and actually calls the instantiated subclass method, or appropriate subclass method.
instanceof	The operator that allows you to compare a class instance against a class type.
casting	The process of explicitly changing one data type to another data type.
abstract class	A class with an abstract constructor and at least one method that is defined but not implemented.
upward cast	This type of casting changes a specialized object instance into a generalized instance. It doesn't lose any of its detail but you can't access them without downcasting the object to access specialized methods.
abstract constructor	A constructor without implementation that makes the class restricted in that it cannot create instances.

Try It/Solve It:

1. Give one reason why you might use an Abstract class rather than an Interface

2. Given the following classes.

```
public class Animal {

        public void makeNoise() {

                System.out.println("talk");

        }
```

```
}
public class Dog extends Animal {
	public void makeNoise() {
		System.out.println("Bark");
	}
}
```

3. What would the output of the following be? Explain your answer.

```
Animal animal = new Animal();
animal.makeNoise();
Dog dog = new Dog();
dog.makeNoise();
Animal animaldog = new Dog();
animaldog.makeNoise();
```

4. Using the animal and dog classes above. If we added the following code to the driver what would the output be:

```
if (animal instanceof Animal)
	System.out.println("animal is Animal");
if (dog instanceof Animal)
	System.out.println("dog is Animal");
if (animaldog instanceof Animal)
	System.out.println("animaldog is Animal");
if (animal instanceof Dog)
	System.out.println("animal is Dog");
```

5. Describe casting.

6. Using the animal and dog classes above. Show examples of using a downcast and an upcast.