**Chapter 9: Polymorphism**

**Multiple Choice Questions**:

1) A polymorphic reference is one that can refer to \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ type(s) of object(s).

a) exactly one

b) zero

c) multiple

d) abstract

e) static

Answer: c

Explanation: A polymorphic reference can point to multiple types of objects at different points in time.

2) The commitment to execute certain code to carry out a method invocation is referred to as \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

a) execution

b) binding

c) polymorphism

d) inheritance

e) none of the above

Answer: b

Explanation: Binding refers to the commitment to execute certain code to carry out a method invocation.

3) In Java, polymorphic method binding occurs \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ .

a) at run time

b) at compile time

c) never

d) when a programmer writes the code

e) during the testing phase of software development

Answer: a

Explanation: polymorphic method binding occurs at run-time.

4) Late binding is \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ than \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ .

a) more efficient, compile-time binding

b) less efficient, compile-time binding

c) more efficient, run-time binding

d) less efficient, run-time binding

e)

Answer: b

Explanation: Late binding is less efficient than compile-time binding due to the overhead associated with determining the code that should be executed at run time.

5) Suppose that Horse is a subclass of Animal, and neither class is abstract. Which of the following is an invalid declaration and initialization?

a) Horse h = new Horse();

b) Horse h = new Animal();

c) Animal a = new Animal();

d) Animal a = new Horse();

e) all of the above are valid

Answer: b

Explanation: Since Horse is a subclass of Animal, choice b would require an explicit cast in order to be valid.

6) In Java, a(n) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ is a collection of constants and abstract methods.

a) polymorphic reference

b) abstract class

c) implementation

d) interface

e) iterator

Answer: d

Explanation: An interface is a collection of constants and abstract methods.

7) In Java, polymorphic references can be created through the use of \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ and \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

a) inheritance, interfaces

b) inheritance, abstract classes

c) interfaces, abstract classes

d) interfaces, iterators

e) none of the above

Answer: a

Explanation: In Java, polymorphic references can be created through the use of inheritance and interfaces.

8) Let Dog be a subclass of Animal, and suppose Animal has a method called speak() that is overridden in the Dog class. Consider the following code.

Animal spot = new Dog();

spot.speak();

Which of the following is true?

a) This code will result in a compile-time error.

b) This code will result in a run-time error.

c) The speak method defined in the Animal class will be called.

d) The speak method defined in the Dog class will be called.

e) The speak method will not be called at all.

Answer: d

Explanation: The speak method defined in the Dog class will be called in this case. At run-time, the Java virtual machine determines that spot is pointing to an object of type Dog and binds the method to the methods defined in the Dog class.

9) The Comparable interface contains which of the following methods?

a) isGreaterThan

b) isLessThan

c) equals

d) compareTo

e) all of the above

Answer: d

Explanation: The Comparable interface contains exactly one method -- compareTo.

10) Let Object a be larger than Object b. What will the following method call return?

a.compareTo(b)

a) it will return 0

b) it will return a number greater than 0

c) it will return a number less than 0

d) it will return true

e) it will return false

Answer: b

Explanation: The compareTo method returns an integer. If Object a is bigger than Object b, it will return a number greater than 0.If Object a is less than Object b, it will return a number less than 0. If they are equal it return 0.

11) Which of the following methods are included with any object that implements the Iterator interface?

a) next

b) hasNext

c) toString

d) all of the above

e) a and b

Answer: d

Explanation: The Iterator interface specifies that all objects that implement it must have the hasNext and next methods. Since all objects in Java are a subclass of the Object class, it will also include the toString method.

12) You need to create a reference variable that can refer to objects from many different classes. You do not know the inheritance hierarchies of the classes. The safest class to use to declare the reference variable is

a) Animal

b) String

c) Object

d) Scanner

e) File

Answer: c

Explanation: All classes are descendants of the Object class, so every object will have the Object class in its inheritance hierarchy.

13) Consider the following line of code.

Comparable s = new String();

Which of the following statements is true about this line?

a) It will result in a compile-time error.

b) It will result in a run-time error.

c) It will create a String object pointed to by a Comparable reference.

d) Although it is perfectly valid Java, it should be avoided due to confusion.

e) none of the above are true

Answer: c

Explanation: This is a valid Java statement and will result in no errors, since the String class implements the Comparable interface.

14) Suppose Animal is an interface that specifies a single method – speak. Now suppose the Dog class implements the Animal interface. In addition to the speak method, the Dog class also has a method called wagTail. Now consider the following code.

Animal a = new Dog();

a.wagTail();

Which of the following is true about this code?

a) It will result in a compile-time error.

b) It will result in a run-time error.

c) It will call the speak method defined in the Animal interface.

d) It will call the wagTail method defined in the Dog class.

e) none of the above are true.

Answer: a

Explanation: This code will result in a compile-time error since the Animal interface does not specify a wagTail method. This compile-time error can be avoided by explicitly casting a as a Dog when calling the wagTail method.

15) Which GUI concepts use polymorphism to establish their relationship?

a) a listener and its associated component

b) a radio button and its default selection

c) a button and its label

d) a slider and its tick marks

e) none of the above

Answer: a

Explanation: Polymorphism is used to establish the relationship between a listener and its associated component.**True/False Questions**:

1) Consider a reference declared in the following manner.

Animal a;

This reference may only point to an object that created by instantiating the Animal class.

Answer: False

Explanation: This reference may point to an object of any type that is compatible with Animal. In particular, it may point to any object that is an instance of a class that is a subclass of Animal.

2) Let Animal be an interface. Then it is possible to create an object by instantiating the Animal interface.

Answer: False

Explanation: An interface cannot be instantiated.

3) The compareTo method of the Comparable interface returns a boolean value.

Answer: False

Explanation: The compareTo method returns an integer.

4) Compile-time binding is more efficient than dynamic binding

Answer: True

Explanation: Binding is the process of connecting method code to a method invocation/call. Compile-time binding occurs when the code is compiled. The connection is made once. Dynamic binding occurs when the program runs. For each call, the program must determine which code to use, and then execute the code. The need to determine the code to invoke for each call during program execution makes dynamic binding slightly less efficient than compile-time binding.

5) A parameter to a method can be polymorphic.

Answer: True

Explanation: A method can accept a polymorphic reference; this may give the method more flexibility than it would otherwise have.

6) An interface cannot declare any instance variables.

Answer: True

Explanation: An interface may declare constants, but it may not declare instance variables.

7) Establishing the relationship between a listener and the component it listens to is accomplished using polymorphism.

Answer: True

Explanation: The association between a component and its listener is performed by using the component’s addActionListener method. The parameter to addActionListener is a reference to an object of the ActionListener interface. The listener object must implement the ActionListener interface and provide a body for the performAction method, which the component invokes when an event occurs.

8) A reference variable can refer to an object of a child class, but not any further down the inheritance hierarchy.

Answer: False

Explanation: A reference variable can refer to an object of any class that is a descendent of the class of the reference variable.

9) Polymorphism via inheritance requires that all classes in the inheritance hierarchy are concrete.

Answer: False

Explanation: A reference variable can be declared at the level of an abstract class. Objects that it refers to must be instantiated from concrete classes.

10) An interface name may be used as a reference type.

Answer: True

Explanation: An interface name may be used a reference type in the same way that a class name may be used as a reference type. Like an abstract class, an interface cannot be instantiated, however. **Short Answer Questions**:

1) What is polymorphism?

Answer: A polymorphic reference is a single type of reference that can point to different types of objects at different times. Polymorphism refers to the fact that the method calls on polymorphic references are selected by the object type, not the reference type. This is often referred to as *late binding*.

2) How does inheritance relate to polymorphism in Java?

Answer: Inheritance is one process by which polymorphic references are created. A reference to a type higher in an inheritance hierarchy can always refer to an object of a type lower in an inheritance hierarchy due to the *is-a* relationship.

3) Consider a class hierarchy that includes a class called Vehicle, with subclasses called Car and Airplane. The Vehicle class has a method called getMaxSpeed, which is overridden in the Car class. The getMaxSpeed of the Vehicle class returns 760 mph, while the getMaxSpeed method of the Car class is overridden to return 150 mph. What is the output of the following snippet of code? Explain your answer.

Vehicle v = new Car();

System.out.println(v.getMaxSpeed() + “ mph”);

Answer: The output of this code will be “150 mph”. Even though the reference is to the Vehicle class, the getMaxSpeed method is bound to the definition in the Car class, since the object is a car. This is due to the polymorphic nature of the reference.

4) Consider the following inheritance hierarchy that is used in a video game.

Character

/ \

Friend Villain

/ \ / \

WiseMan ShopKeeper Dragon Skeleton

| |

FlyingDragon EliteSkeleton

Which of the following declarations and initializations will not cause a compiler error?

Character c = new FlyingDragon();

FlyingDragon f = new Character();

Dragon d = new Villain();

Villain v = new Skeleton();

Dragon d = new ShopKeeper();

Answer: The following are valid for this inheritance hierarchy because of the *is-a* relationship.

Character c = new FlyingDragon();

Villain v = new Skeleton();

5) When a reference variable refers to an object that is in an inheritance hierarchy and a method of the object is invoked, how does Java determine which version of the method to use?

Answer: Java determines the definition of the method that is closest to the object in the inheritance hierarchy between the reference variable and the object.

6) Are there any differences between extending a class and implementing an interface?

Answer: Yes. First, extending a class uses the extends keyword, while implementing an interface uses the implements keyword. More importantly, a class can implement multiple interfaces, while it can only extend a single class.

7) Describe the compareTo method and the circumstances under which it returns different values.

Answer: The compareTo method is specified by the Comparable interface. It places an ordering on objects. Consider the following call to compareTo:

int result = obj1.compareTo(obj2);

In this case result will be positive if obj1 is larger than obj2 in the sense of the ordering. It will be 0 if obj1 and obj2 are the same, and it will be negative if obj2 is larger than obj1.

8) Does polymorphism work if some of the classes in an inheritance hierarchy are abstract?

Answer: Yes. A reference variable can be declared at any level in the inheritance hierarchy. If it is declared to be of an abstract class, it can still be used to refer to objects of concrete classes.

9) Can an interface hierarchy be used for polymorphism? Explain.

Answer: Yes. A reference variable can use an interface as its type, and can then refer to objects of classes that implement the interface. In this way, polymorphism works the same as it does with an inheritance hierarchy.

10) Write an interface for a CD player. It should have the standard operations (i.e. play, stop, etc) that usual CD players have.

Answer:

public interface CDPlayer {

public void play();

public void stop();

public void nextTrack();

public void previousTrack();

public void seekForward();

public void seekBackwards();

}

11) Give an example of a class that implements the Comparable interface, and explain how the implementation of the compareTo method determines its return value.

Answer: The String class implements the Comparable interface. The comparison is based on the lexicographic ordering of String objects defined by the Unicode character set.

12) Is dynamic binding used when polymorphic references are made using interfaces?

Answer: Yes. Java must use the type of the object to identify which method code to use for an invocation. This identification is done at execution-time, not at compile time, and is therefore considered dynamic binding.

13) Can a polymorphic reference invoke a method that is only declared at the object’s class level? If “yes”, explain how.

Answer: In general, a polymorphic reference can only be used to invoke methods that are known to the class of the reference variable. In order to call a method that is declared at the object’s class level, the reference variable must be cast to be of the object’s type as part of the call.

14) Suppose you are implementing the comparable interface in a class representing a Person, where the ordering is based on the age of the person. Write a compareTo method for this class. You may assume that there is an instance variable called age and an accessor method called getAge.

Answer:

public int compareTo(Object o) {

Person p = (Person) o;

return this.getAge() - p.getAge();

}

15) Why can't an interface be instantiated?

Answer: An interface cannot be instantiated because, similar to an abstract class, it only contains abstract methods.