**Chapter 8: Inheritance**

**Multiple Choice Questions**:

1) The process of inheritance should establish a(n) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ relationship.

a) is-a

b) has-a

c) static

d) not-a

e) none of the above

Answer: a

Explanation: Inheritance should establish an *is-a* relationship. Therefore any objects that are of a type lower in the inheritance hierarchy are also of a type higher in the inheritance hierarchy.

2) The original class that is used to derive a new class using inheritance is called \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ .

a) a superclass

b) a parent class

c) a base class

d) all of the above

e) neither a, b, nor c

Answer: d

Explanation: The original class can be called a superclass, a parent class, or a base class.

3) \_\_\_\_\_\_\_\_\_\_ occurs when a child class defines a method with the same signature as a method in the parent class.

a) Overloading

b) Overriding

c) Overwhelming

d) Substituting

e) A child class cannot define a method with the same signature as a parent class method.

Answer: b

Explanation: Overriding occurs when two methods, one in the parent class and one in the child class, have the same signature. The signature is the method name and the parameter list.

4) In order for derived classed to have access to encapsulated data members and methods of superclasses, the data members and methods should be declared using the \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ modifier.

a) private

b) public

c) protected

d) final

e) static

Answer: c

Explanation: Data members and methods declared using the protected modifier can be accessed by subclasses in an inheritance hierarchy, but are still encapsulated from classes and methods outside of the hierarchy.

5) A child class can access private members of a parent class by

a) using super in front of the member name

b) using the member name directly

c) using this in front of the member name

d) using the public accessor and mutator methods defined in the parent class

e) A child class cannot access private members of a parent class.

Answer: e

Explanation: Private members of a parent class cannot be accessed from an object of a child class directly or through the use of super. They can be accessed by using the public accessor and mutator methods that the parent class provides.

6) When a variable declared in a subclass has the same name as a variable declared in a superclass, it is called a \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ variable.

a) final

b) shadow

c) static

d) dead

e) this is not allowed in Java

Answer: b

Explanation: A shadow variable is a variable in a subclass with the same name as a variable in the superclass.

7) A(n)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ class represents a generic concept in a class hierarchy.

a) super

b) abstract

c) interface

d) shadow

e) generic

Answer: b

Explanation: An abstract class represents a generic entity that is not completely defined. An abstract class cannot be instantiated. It contains one or more abstract methods, which are methods that should be overridden by subclasses.

8) A class declared as final \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ .

a) cannot be changed.

b) cannot have subclasses.

c) cannot have superclasses.

d) has several abstract methods.

e) cannot be used in a program.

Answer: b

Explanation: The final modifier restricts inheritance. In particular, a class declared as final cannot have subclasses.

9) Which of the following key words indicates a method that cannot be overridden in a derived class?

a) super

b) final

c) extends

d) inherits

e) expands

Answer: b

Explanation: The key word final, when used in a method header, indicates that the method cannot be overridden in a derived class.

10) To invoke a parent’s constructor in a subclass, we use the \_\_\_\_\_\_\_\_\_\_\_\_\_\_ method.

a) abstract

b) construct

c) parent

d) super

e) extends

Answer: d

Explanation: The super method is used to invoke a parent’s constructor from a subclass.

11) Which of the following statements is *not* a general inheritance practice that you should keep in mind in the design of a program?

a) Derived classes should have an “is-a” relationship with the parent classes.

b) Use the final key word when defining parent classes.

c) Avoid shadowing inherited variables when possible.

d) Define abstract classes to specify a common class interface for concrete derived classes.

e) All of these are general inheritance practices that should be considered when designing a program.

Answer: b

Explanation: Using the final key word in a class header indicates that the class cannot be extended, or inherited from. It cannot be used in a class header if the class is intended to be the parent of one or more classes.

12) All Java classes are subclasses of the \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ class.

a) String

b) java.lang

c) Java

d) Class

e) Object

Answer: e

Explanation: All classes are subclasses of Java's Object class, whether explicitly specified or not.

13) When designing a class hierarchy, it is important that common features be \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ .

a) higher in the class hierarchy.

b) lower in the class hierarchy.

c) near the middle of the class hierarchy.

d) in abstract classes.

e) in the Object class.

Answer: a

Explanation: Common features should be included closer to the top of the class hierarchy. Doing this makes them available to more classes lower in the hierarchy.

14) Which of the following methods are included in every class created in Java by inheritance?

a) next

b) toString

c) compareTo

d) charAt

e) none of the above

Answer: b

Explanation: The toString method is defined in the Object class, and is therefore included in every Java class via inheritance.

15) Of the classes below, the one that is most likely to be declared abstract is \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

a) Bat

b) Squirrel

c) Animal

d) Iguana

e) Parrot

Answer: c

Explanation: The Animal class is most likely to be abstract since it is the most generic.**True/False Questions**:

1) A parent class object must be created before objects of a child class can be created.

Answer: False

Explanation: Objects can be created from concrete classes at any point of an inheritance hierarchy in any order.

2) Private members of a parent class are inherited by child classes.

Answer: True

Explanation: Child classes inherit all of the members of a parent class, whether they are public, private, or protected. Private members cannot be accessed directly (by name) in the child class, but they are part of the derived class.

3) Java supports multiple inheritance.

Answer: False

Explanation: Java does not support true multiple inheritance, but it is possible to get some of the features of multiple inheritance using interfaces.

4) In Java, a subclass can only extend one parent class.

Answer: True

Explanation: Allowing a subclass to extend multiple parent classes leads to multiple inheritance, which is not supported in Java.

5) A child class is allowed to define a method with the same name and parameter list as a method in the parent class.

Answer: True

Explanation: A subclass is allowed to override methods that are in the parent class.

6) A child class is allowed to declare a variable with the same name as one that is contained in the parent class.

Answer: True

Explanation: This is known as *variable shadowing* and can lead to confusion. It is, however, permitted in Java.

7) An abstract class must contain abstract methods.

Answer: False

Explanation: A class declared as abstract may or may not contain abstract methods.

8) It makes sense to declare most abstract classes as final.

Answer: False

Explanation: Since an abstract class cannot be instantiated, it makes no sense to declare it as final. It is usually expected that an abstract class will be extended.

9) It is possible to derive a class from an abstract class without overriding all of the parents abstract methods.

Answer: True

Explanation: The child class must also be declared as abstract in this case.

10) Inheritance should not be considered in the software design process.

Answer: False

Explanation: Inheritance should be carefully considered in the software design process. Software systems designed carefully using inheritance can be more flexible than software designed without considering inheritance.**Short Answer Questions**:

1) Explain why inheritance is useful.

Answer: Inheritance is useful because it allows for code-reuse. This means that if we have multiple software entities that have common features, the code for the common features can be written once in a superclass. The classes that include these features can then be written via inheritance, and this common code does not have to be rewritten.

2) Compare and contrast the private visibility modifier to the protected visibility modifier. Why is the protected visibility modifier a better choice in an inheritance hierarchy?

Answer: The private modifier and the protected modifier both enforce the encapsulation of instance variables and methods in a class. This means that unrelated classes are not able to directly access the variables and the methods. The protected modifier, however, *does* allow for the instance variables and the methods to be accessed by subclasses of the original class. This makes the protected modifier a better choice for use in an inheritance hierarchy, because it is often useful for subclasses to have access to a superclass's instance variables and methods.

3) Suppose we create a subclass from a class that has a method called someMethod. If we override someMethod in the subclass, is it possible to access the superclass's version of someMethod? If so, how?

Answer: Yes, it is possible to access the original version of the method. To do so, we qualify the call to the method with the super reference. In other words, to access the original version of the method we call super.someMethod().

4) Can a class be a parent of more than one subclass? Can a class be a child of more than one parent? Explain.

Answer: A class can be the parent of more than one subclass in an inheritance hierarchy. Classes that have the same parent class are often called *siblings*. In Java, a class cannot be the child of more than one parent since Java does not support multiple inheritance.

5) Explain the relevance of the Object class to the Java programming language.

Answer: Every class in Java is a subclass of the Object class. This occurs whether a class definition explicitly extends the Object class or not. Therefore, every class in Java has a common set of methods that are defined in the Object class. These include the toString method and the equals method.

6) What is an abstract class, and why might it be useful in an inheritance hierarchy?

Answer: An abstract class is a class represents a partially defined concept in an inheritance hierarchy. Abstract methods cannot be instantiated, but they can be extended. They are often useful because several classes may include common functionality but may lack a fully defined parent concept. Abstract classes allow a programmer to implement the partially defined parent concept as an abstract class which will include the common functionality of the child classes. An example of an abstract concept in an inheritance hierarchy might be a Vehicle. Subclasses like Car, Boat and Airplane are more fully defined, but they share common states and behaviors.

7) Explain how a subclass can can access its parent classes private instance variables and methods.

Answer: A subclass can access private instance variables and methods of its parent class, but only indirectly. There must be public methods that access the private data and methods directly. These public methods can then be called by the subclass, which gives indirect access to the private variables and methods of the parent class.

8) A programmer tries to create a subclass of String called MyString. When the programmer compiles her new class, the compiler produces the following message:

MyString.java:1: cannot inherit from final java.lang.String

public class MyString extends String {

^

1 error

Explain the cause of this error.

Answer: The String class has been declared with the final modifier, which restricts any other classes from extending it.

9) Draw a hierarchy of Animals. The hierarchy should include the following entities: Animal, Reptile, Mammal, Bear, Human, Iguana, and Dolphin. Note that an Iguana is a Reptile, a Bear is a Mammal, a Human is a Mammal, and a Dolphin is a Mammal.

Answer:

Animal

/ \

Reptile Mammal

/ / | \

Iguana Bear Human Dolphin

10) Consider a software system that will implement the following classes: Student, Professor, StaffMember, ContractWorker. List some common attributes of these classes. What would be a good abstract class from which these classes may be extended via inheritance?

Answer: Some common attributes would be socialSecurityNumber, age, and address. A good abstract class from which these classes could be extended is Person.

11) Explain what it means for a child class to override a method in a parent class. Why might this be useful?

Answer: A child class overrides a method that is inherited from a parent class by redefining it in the subclass. This is useful because a subclass may have a slightly different behavior than the superclass, but the behavior still has the same name. Overriding methods allows for this flexibility.

12) Why is it considered a good practice to override the toString and equals methods?

Answer: These methods are defined in the Object class and are inherited by all classes. By overriding these methods, a programmer can define behavior that more closely matches the purpose of the class, rather than using the default behavior as defined in the Object class.

13) Describe the behavior of the toString method and the equals method of the Object class.

Answer: The toString method returns a String representation of an object. The equals method returns true if the object is an alias of the object sent in as a parameter.

14) What does it mean for a class to be declared as final? What does it mean for a method to be declared as final?

Answer: A method declared as final cannot be overridden by any subclass. A class declared as final can not be extended via inheritance.

15) What is a shadow variable?

Answer: A shadow variable is a variable that is declared in a subclass that has the same name as a variable declared in the class's parent class.