AP Computer Science
Review for Chapter 10 Test

1. Consider the classes shown below:

```
public class Parent
{
   public int getValue()
   {
      return 24;
   }
   public void display()
   {
      System.out.print(getValue() + " ");
   }
}

public class Child extends Parent
{
   public int getValue()
   {
      return -7;
   }
}
```

Using the classes above, what is the output of the following lines of code?

```
Child kid = new Child();
   Parent adult = new Parent();
   kid.display();
adult.display();
```

a) `24 24`
b) `-7 -7`
c) `-7 24`
d) `24 -7`

Answer: c

2. Consider the classes shown below:

```
public class Parent
{
   public int getValue()
   {
      return 24;
   }
   public void display()
   {
      System.out.print(getValue() + " ");
   }
}

public class Child extends Parent
{
```

```
   public int getValue()
   {
      return -7;
   }
}
```

Using the classes above, what is the output of the following lines of code?

```
   Parent kid = new Child();
   Parent adult = new Parent();
   kid.display();
   adult.display();
```

a) `-7 24`
b) `24 24`
c) `-7 -7`
d) `24 -7`

Answer: a

3. Suppose the abstract class `Message` is defined below

```
public abstract class Message {
   private String value;

   public Message(String initial)
   {
      value = initial;
   }

   public String getMessage()
   {
      return value;
   }

   public abstract String translate();
}
```

A concrete subclass of `Message`, `FrenchMessage`, is defined. Which methods must `FrenchMessage` define?

a) `translate()` only
b) `getMessage()` only
c) The `FrenchMessage` constructor and `translate()` only
d) The `FrenchMessage` constructor, `getMessage()`, and `translate()`

Answer: a

4. Consider the following class hierarchy:

```
public class Vehicle
{
```

```
   private String type;
   public Vehicle(String type)
   {
      this.type = type;
   }
   public String getType()
   {
      return type;
   }
}

public class LandVehicle extends Vehicle
{
   public LandVehicle(String type)
   {
      . . .
   }
}

public class Auto extends LandVehicle
{
   public Auto(String type)
   {
      . . .
   }
}
```

Which of the following code fragments is NOT valid in Java?

a) `Vehicle myAuto = new Auto("sedan");`
b) `LandVehicle myAuto = new Auto("sedan");`
c) `Auto myAuto = new Auto("sedan");`
d) `LandVehicle myAuto = new Vehicle("sedan");`

Answer: d

5. Consider the following code snippet:

```
Vehicle aVehicle = new Auto();
aVehicle.moveForward(200);
```

If the `Auto` class inherits from the `Vehicle` class, and both classes have an implementation of the `moveForward` method with the same set of parameters and the same return type, which statement is correct?

a) The `moveForward` method of the `Auto` class will be executed.
b) The `moveForward` method of the `Vehicle` class will be executed.
c) You must specify in the code which class's `moveForward` method is to be used.
d) It is not possible to determine which class's method is called.

Answer: a

6. Consider the following code snippet:

```
Employee anEmployee = new Programmer();
anEmployee.increaseSalary(2500);
```

If the `Programmer` class inherits from the `Employee` class, and both classes have an implementation of the `increaseSalary` method with the same set of parameters and the same return type, which statement is correct?

a) The `increaseSalary` method of the `Programmer` class will be executed.
b) The `increaseSalary` method of the `Employee` class will be executed.
c) You must specify in the code which class's `increaseSalary` method is to be used.
d) It is not possible to determine which class's method is called.

Answer: a

7. Consider the following code snippet:

```
Vehicle aVehicle = new Auto();
aVehicle.moveForward(200);
```

Assume that the `Auto` class inherits from the `Vehicle` class, and both classes have an implementation of the `moveForward` method with the same set of parameters and the same return type. The process for determining which class's `moveForward` method to execute is called ____.

a) inheritance disambiguation.
b) inheritance hierarchy.
c) dynamic inheritance.
d) dynamic lookup.

Answer: d

8. Consider the following code snippet:

```
Vehicle aVehicle = new Auto();
aVehicle.moveForward(200);
```

Assume that the `Auto` class inherits from the `Vehicle` class, and both classes have an implementation of the `moveForward` method with the same set of parameters and the same return type. Which class's `moveForward` method is to be executed is determined by ____.

a) the actual object type.
b) the variable's type.
c) the hierarchy of the classes.
d) it is not possible to determine which method is executed.

Answer: a

9. Consider the following code snippet:

```
Employee anEmployee = new Programmer();
anEmployee.increaseSalary(2500);
```

Assume that the `Programmer` class inherits from the `Employee` class, and both classes have an implementation of the `increaseSalary` method with the same set of parameters and the same return type. Which class's `increaseSalary` method is to be executed is determined by ____.

a) the hierarchy of the classes.
b) the variable's type.
c) the actual object type.
d) it is not possible to determine which method is executed.

Answer: c

10. Which of the following statements about abstract methods is true?

a) An abstract method has a name, parameters, and a return type, but no code in the body of the method.
b) An abstract method has parameters, a return type, and code in its body, but has no defined name.
c) An abstract method has a name, a return type, and code in its body, but has no parameters.
d) An abstract method has only a name and a return type, but no parameters or code in its body.

Answer: a

11. Which of the following statements about classes is true?

a) You can create an object from a concrete class, but not from an abstract class.
b) You can create an object from an abstract class, but not from a concrete class.
c) You cannot have an object reference whose type is an abstract class.
d) You cannot create subclasses from abstract classes.

Answer: a

12. If a class has an abstract method, which of the following statements is NOT true?
a) You can construct an object from this class.
b) You can have an object reference whose type is this class.
c) You can inherit from this class.
d) All non-abstract subclasses of this class must implement this method.

Answer: a

13. Consider the following code snippet:

```
public abstract class Machine
{
   public abstract void setRPMs();
   . . .
}
```

You wish to create a concrete subclass named `PolisherMachine`. Which of the following is the correct way to declare this subclass?

a)
```
public class PolisherMachine implements Machine
{
   public void setRPMs() { . . . }
}
```
b)
```
public class PolisherMachine extends Machine
{
   void setRPMs() { . . . }
}
```

c)
```
public class PolisherMachine implements Machine
{
   void setRPMs() { . . . }
}
```

d)
```
public class PolisherMachine extends Machine
{
   public void setRPMs() { . . . }
}
```

Answer: d

14. Consider the following code snippet:

```
public abstract class Employee
{
   public abstract void setSalary();
   . . .
}
```

You wish to create a concrete subclass named `Programmer`. Which of the following is the correct way to declare this subclass?

a)
```
public class Programmer implements Employee
{
   public void setSalary() { . . . }
}
```
b)
```
public class Programmer extends Employee
{
   void setSalary() { . . . }
}
```

c)
```
public class Programmer implements Employee
{
   void setSalary() { . . . }
}
```

d)

```
public class Programmer extends Employee
{
   public void setSalary() { . . . }
}
```

Answer: d

15. A class from which you cannot create objects is called a/an ____.

a) Abstract class.
b) Concrete class.
c) Non-inheritable class.
d) Superclass.

Answer: a

16. A class that cannot be instantiated is called a/an ____.

a) Abstract class.
b) Anonymous class.
c) Concrete class.
d) Non-inheritable class.

Answer: a

17. Which of the following statements about classes is true?

a) You can create an object from a class declared with the keyword `final`.
b) You can override methods in a class declared with the keyword `final`.
c) You can extend a class declared with the keyword `final`.
d) You can create subclasses from a class declared with the keyword `final`.

Answer: a

18. The ____ reserved word in a class definition ensures that subclasses cannot be created from this class.

a) `abstract`
b) `anonymous`
c) `final`
d) `static`

Answer: c

19. The ____ reserved word in a method definition ensures that subclasses cannot override this method.

a) `abstract`
b) `anonymous`
c) `final`
d) `static`

Ans: c

20. Which of the following is true regarding inheritance?

a) When creating a subclass, all methods of the superclass must be overridden.
b) When creating a subclass, no methods of a superclass can be overridden.
c) A superclass can force a programmer to override a method in any subclass created from it.
d) A superclass cannot prevent a programmer from overriding a method in any subclass created from it.

Answer: c

21. When declared as `protected`, data in an object can be accessed by ____.

a) Only by that class's methods and by all of its subclasses
b) Only by that class's methods, by all of its subclasses, and by methods in classes within the same package.
c) Only by that class's methods.
d) By any class.

Answer: b

22. Consider the following code snippet:

```
public class Vehicle
{
   protected int numberAxles;
   . . .
}
```

Data in the `numberAxles` variable can be accessed by ____.

a) Only by the `Vehicle` class's methods and by all of its subclasses
b) Only by the `Vehicle` class's methods, by all of its subclasses, and by methods in classes within the same package.
c) Only by the `Vehicle` class's methods.
d) By any class.

Answer: b

23. To ensure that an instance variable can only be accessed by the class that declared it, the variable should be declared as ____.

a) `public`

b) `private`
c) `protected`
d) `final`

Answer: b

24. With a few exceptions, instance variables of classes should always have ___ access.

a) `final`
b) `private`
c) `public`
d) `protected`

Answer: b

25. Consider the following code snippet:

```
double salary = 45000.00;
String sal = "Current salary is " + salary;
```

Which of the following statements is correct?

a) The `toString()` method of the `Object` class is being used to set the value of `salary`.
b) The `toString()` method of the `Double` class is being used to set the value of `salary`.
c) No `toString()` method is being used to set the value of `salary`.
d) This code will not compile.

Answer: c

Title: Which statement about this code is correct?
Difficulty: Medium
Section Reference: 9.5 `Object`: The Cosmic Superclass
Section Reference 2: 9.5.1 Overriding the `toString` method

26. Consider the following code snippet:

```
Auto consumerAuto = new Auto(4, "gasoline");
String s = consumerAuto.toString();
```

Assume that the `Auto` class has not implemented its own `toString()` method. What value will `s` contain when this code is executed? a) `s` will contain the values of the instance variables in `consumerAuto`.
b) `s` will contain only the class name of the `consumerAuto` object.
c) `s` will contain the class name of the `consumerAuto` object followed by a hash code.
d) This code will not compile.

Answer: c

27. Consider the following code snippet of a function object

```
public interface Measurer
{
   double measure(______ anObject);
}
```

Complete this code to allow the interface to handle all classes?

a) `Class`
b) `Object`
c) `Any`
d) `Void`

Answer: b

28. Consider the following code snippet:

```
if(anObject instanceof Auto)
{
   Auto anAuto = (Auto) anObject;
   . . .
}
```

What does this code do?

a) This code tests whether `anObject` was created from a superclass of `Auto`.
b) This code creates a subclass type object from a superclass type object.
c) This class safely converts an object of any type to an object of type `Auto`.
d) This code safely converts an object of type `Auto` or a subclass of `Auto` to an object of type `Auto`.

Answer: d

29. To test whether an object belongs to a particular type, use ___.

a) the `this` reserved word.
b) the `subclassOf` reserved word.
c) the `instanceof` operator.
d) the `equals` method.

Answer: c