1. In an inheritance relationship, this is the general class.

a. subclass

b. superclass

c. slave class

d. child class

2. In an inheritance relationship, this is the specialized class.

a. superclass

b. master class

c. subclass

d. parent class

3. This key word indicates that a class inherits from another class.

a. derived

b. specialized

c. based

d. extends

4. A subclass does not have access to these superclass members.

a. public

b. private

c. protected

d. all of these

5. This key word refers to an object's superclass.

a. super

b. base

c. superclass

d. this

6. In a subclass constructor, a call to the superclass constructor must _____.

a. appear as the very first statement

b. appear as the very last statement

c. appear between the constructor's header and the opening brace

d. not appear

7. The following is an explicit call to the superclass's default constructor.

a. default();

b. class();

c. super();

d. base();

8. A method in a subclass that has the same signature as a method in the superclass is an example of _____.

a. overloading

b. overriding

c. composition

d. an error

9. A method in a subclass having the same name as a method in the superclass but a different signature is an example of _____.

a. overloading

b. overriding

c. composition

d. an error

10. These superclass members are accessible to subclasses and classes in the same package.

a. private

b. public

c. protected

d. all of these

11. All classes directly or indirectly inherit from this class.

a. Object

b. Super

c. Root

d. Java

12. With this type of binding, the Java Virtual Machine determines at runtime which method to call, depending on the type of the object that a variable references.

a. static

b. early

c. flexible

d. dynamic

13. This operator can be used to determine whether a reference variable references an object of a particular class.

a. isclass

b. typeof

c. instanceof

d. isinstance

22. **True** or False: Constructors are not inherited.

23. **True** or False: In a subclass, a call to the superclass constructor can only be written in the subclass constructor.

24. True or **False**: If a subclass constructor does not explicitly call a superclass constructor, Java will not call any of the superclass's constructors.

25. True or **False**: An object of a superclass can access members declared in a subclass. 26. **True** or False: The superclass constructor always executes before the subclass constructor.

27. True or **False**: When a method is declared with the final modifier, it must be overridden in a subclass.

28. True or **False:** A superclass has a member with package access. A class that is outside the superclass's package but inherits from the superclass can access the member.

29. **True** or False: A superclass reference variable can reference an object of a subclass that extends the superclass.

30. True or **False**: A subclass reference variable can reference an object of the superclass.

Algorithm Workbench

1.

```
public class Poodle extends Dog{

}
```

2. Look at the following code, which is the first line of a class definition:

public class Tiger extends Felis

In what order will the class constructors execute?

The super class executes before the subclass, so Felis would executes first and then Tiger.

3.

Assignment2 > classB.java > classB > calc()

```java
package Assignment2;

public abstract class classB {

    private int m;
    protected int n;


    public void setM(int m){
        this.m = m;
    }
    public int getM(){
        return m;
    }

    public void setN(int n){
        this.n = n;
    }
    public int getN(){
        return n;
    }

    public abstract double calc();

}

```

Assignment2 > classD.java > classD > calc()

```java
package Assignment2;

public class classD extends classB{

    private double q;
    protected double r;

    public void setQ(int q){
        this.q = q;
    }
    public double getQ(){
        return q;
    }

    public void setR(int r){
        this.r = r;
    }
    public double getR(){
        return r;
    }

    @Override
    public  double calc(){
        return q * r;
    }


}

```

4.

Assignment2 > XYZ.java > ...

```java
package Assignment2;

public class XYZ {

    XYZ(int x , int y, int z){
        super(x, y, z);
    }

}

```

5.

Assignment2 > J setValue.java > ...

```java
package Assignment2;

public class setValue extends Value{

    setValue(){
        setV(v: 10);
    }

}

```

6.

Assignment2 > J gsValue.java > ...

```java
package Assignment2;

public class gsValue extends gValue{

    private int value;

    @Override
    public int getValue() {
        return value;
    }

}

```