1.Has-a Relationship: Represents composition a class contains another class as a field or member.

Is-a Relationship: Represents inheritance a subclass inherits from a parent class.

2. Both go() (from the base class) and stop() (from the derived class) will be available to an object of the derived class because the derived class inherits all public methods from the base class while also defining its own methods.

3. Implementing an Abstract Method: Abstract methods are declared in an abstract class or interface and must be implemented in a derived class.The method has no body in the abstract class or interface, forcing the derived class to provide its own implementation.

Overriding a Method: Occurs when a subclass provides a specific implementation of a method already defined in a parent class.The method in the parent class has a body, but the derived class can redefine it to change or extend its behavior.

4.

Abstract Class:Can contain both abstract methods (no body) and concrete methods (with implementation).Allows inheritance but supports only single inheritance.Can have instance variables and constructors.

Interface:Contains only abstract methods (before Java 8) or may have default/static methods (from Java 8 onward).Allows multiple inheritance, as a class can implement multiple interfaces.Cannot have instance variables (only static and final constants).

6.

a) doThat() is a public abstract method because it is defined in the Wo interface without a body.

b) Wo is an interface because it is declared with the interface keyword.

c) doThat() is implemented in Roo because Roo implements the Wo interface, and all methods in an interface must be implemented by the implementing class unless the class is abstract.

d)

1. doThis() (overrides the method from Bo).
2. doThat() (implementation of the method from Wo).
3. doNow() (inherited from Bo).
4. toString() (inherited from Object).

Here are the answers for questions 6e, 6f, 6g, and 6h:

6e) The implementation of doThis() in Roo overrides the method doThis() in Bo. When doThis() is called on a Roo object, the overridden version in Roo is executed instead of the one in Bo.

6f)The statement super(1); in the constructor of Roo calls the constructor of the parent class Bo with the argument 1. It initializes the fields or properties of the parent class Bo.

6g) No, the doThis() method in Bo cannot be called directly from a Roo object because Roo overrides this method with its own implementation. However, the overridden method in Bo can be called indirectly by using super within Roo:

```
public int callBoDoThis() {
    return super.doThis(); // Calls the implementation of doThis() in Bo
}
```

6h) Yes, a method in Roo can call the doThis() method in Bo by using the super keyword:

```
public int callBoDoThis() {
    return super.doThis(); // Calls the overridden method in Bo
}
```

This allows the Roo class to access the parent implementation of doThis().