1. Explain the difference between a has-a and is-a relationship among classes.

An is-a relationship represents inheritance, where a derived class is a specialized form of a base class.
A has-a relationship represents composition, where one class contains an object of another class as a member variable.

2. If a base class has a public method go() and a derived class has a public method stop(), which methods will be available to an object of the derived class?

An object of the derived class will have access to both the inherited go() method from the base class and the stop() method defined in the derived class.

3. Compare and contrast implementing an abstract method and overriding a method.

Implementing an abstract method requires providing a method body for a method that has no implementation in the abstract base class.
Overriding a method involves redefining a method that already has an implementation in the base class in order to change its behavior in the derived class.

4. Compare and contrast an abstract class and an interface.

An abstract class may contain instance variables, constructors, and both abstract and non-abstract methods, and a class may extend only one abstract class.
An interface contains method declarations only, does not contain instance variables, and a class may implement multiple interfaces.

6. Code-based questions
a) What type of method is doThat() in Wo?

doThat() is an abstract method.

b) What is Wo?

Wo is an interface.

c) Why is doThat() implemented in Roo?

Because Roo implements the Wo interface, it is required to provide an implementation for all methods declared in the interface.

d) List the methods available to a Roo object.

The methods available to a Roo object are:

doThis() from Roo

doNow() inherited from Bo

doThat() implemented in Roo

e) How does the implementation of doThis() in Roo affect the implementation of doThis() in Bo?

The doThis() method in Roo overrides the implementation of doThis() in Bo, so the Roo version is executed when called on a Roo object.

f) What action does the statement super(1) in Roo perform?

It calls the constructor of the base class Bo and initializes its instance variable with the value 1.

g) Can the doThis() method in Bo be called from a Roo object? If so, how?

Yes. It can be called using the super.doThis() statement.

h) Can a method in Roo call the doThis() method in Bo? If so, how?

Yes. A method in Roo can call the base class version using super.doThis().