1. Question 1b:

   This code shows how abstract classes and method overriding work in Java. The abstract class Animal defines a general idea of an animal with a name and an abstract method greets() that subclasses must implement. The Cat class overrides greets() to print “Meow,” while the Dog class overrides it to print “Woof” and adds another version that greets another dog with “Woooof.” The BigDog class extends Dog and overrides the greets methods with louder versions of the sounds. In the main method, Cat, Dog, and BigDog objects are created, and their greets methods are called to show how polymorphism and method overriding make each animal behave differently.

2. Question 2b

   This code demonstrates abstraction, inheritance, and type casting in Java using different geometric shapes. The abstract class Shape defines shared properties like color and filled, as well as abstract methods getArea() and getPerimeter() that subclasses must implement. Circle, Rectangle, and Square each extend Shape and provide their own versions of these methods. The main class tests how upcasting (treating a subclass as its parent class) and downcasting (converting back to a subclass) work. It shows that even when a shape is referenced as a parent type, the correct subclass methods still run thanks to polymorphism. The program also illustrates how Square inherits from Rectangle and keeps both sides equal while still being treated as a type of Shape.

3. Question 3a

   This code is all about shapes and how they can share common behaviors. The GeometricObject interface sets the basic rules, saying that every shape needs to know how to calculate its area and perimeter. Then, the Circle and Rectangle classes follow those rules but each in their own way, the circle uses its radius while the rectangle uses its width and length. The toString methods make it easy to print out what each shape looks like. Overall, it shows how interfaces work in Java and how different shapes can be handled in a similar way.

4. Question 3b:

   This code shows how interfaces and inheritance work in Java. The GeometricObject interface defines methods for getting area and perimeter. The Circle class implements this interface and calculates its area and perimeter. The Resizable interface adds a way to resize shapes by a percentage. ResizableCircle extends Circle and implements Resizable, so it can change size while keeping circle behaviors. The TestResizableCircle class creates a resizable circle and shows its area and perimeter before and after resizing. This demonstrates adding new features to existing classes without changing their core design.