Q1.1. Write a program to show Interface Example in java?

Ans -
```
// Define an interface named Animal
interface Animal {
   void sound(); // Abstract method
}

// Implement the Animal interface in the Dog class
class Dog implements Animal {
   public void sound() {
      System.out.println("Woof!");
   }
}

// Implement the Animal interface in the Cat class
class Cat implements Animal {
   public void sound() {
      System.out.println("Meow!");
   }
}

// Main class to test the interface implementation
public class InterfaceExample {
   public static void main(String[] args) {
      Animal dog = new Dog();
      dog.sound(); // Outputs "Woof!"

      Animal cat = new Cat();
      cat.sound(); // Outputs "Meow!"
   }
}
```

Q2.Write a program a Program with 2 concrete method and 2 abstract method in java ?

Ans -
```
// Abstract class with two abstract methods
abstract class Shape {
   // Abstract method to calculate area
   public abstract double calculateArea();

   // Abstract method to calculate perimeter
   public abstract double calculatePerimeter();

   // Concrete method to display the shape's information
   public void display() {
      System.out.println("Shape: " + this.getClass().getSimpleName());
```

```
        System.out.println("Area: " + calculateArea());
        System.out.println("Perimeter: " + calculatePerimeter());
        System.out.println();
    }
}

// Rectangle class that extends the Shape abstract class
class Rectangle extends Shape {
    private double length;
    private double width;

    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }

    // Implementation of calculateArea() method for Rectangle
    public double calculateArea() {
        return length * width;
    }

    // Implementation of calculatePerimeter() method for Rectangle
    public double calculatePerimeter() {
        return 2 * (length + width);
    }
}

// Circle class that extends the Shape abstract class
class Circle extends Shape {
    private double radius;

    public Circle(double radius) {
        this.radius = radius;
    }

    // Implementation of calculateArea() method for Circle
    public double calculateArea() {
        return Math.PI * radius * radius;
    }

    // Implementation of calculatePerimeter() method for Circle
    public double calculatePerimeter() {
        return 2 * Math.PI * radius;
    }
```

```
}

// Main class to test the shape classes
public class ShapeExample {
    public static void main(String[] args) {
        Shape rectangle = new Rectangle(5, 7);
        rectangle.display();

        Shape circle = new Circle(3);
        circle.display();
    }
}
```

Q3.Write a program to show the use of functional interface in java?

Ans -

```
// Define a functional interface with a single abstract method
@FunctionalInterface
interface Calculator {
    int calculate(int a, int b);
}

// Main class to test the functional interface
public class FunctionalInterfaceExample {
    public static void main(String[] args) {
        // Using lambda expression to implement the functional interface
        Calculator addition = (a, b) -> a + b;
        int result1 = addition.calculate(10, 5);
        System.out.println("Addition result: " + result1);

        Calculator subtraction = (a, b) -> a - b;
        int result2 = subtraction.calculate(10, 5);
        System.out.println("Subtraction result: " + result2);

        Calculator multiplication = (a, b) -> a * b;
        int result3 = multiplication.calculate(10, 5);
        System.out.println("Multiplication result: " + result3);
    }
}
```

Q4.What is an interface in Java?
Ans - In Java, an interface specifies the behavior of a class by providing an abstract type. As one of Java's core concepts, abstraction, polymorphism, and multiple inheritance are supported through this technology. Interfaces are used in Java to achieve abstraction.

Q5.What is the use of interface in Java?

Ans - Interfaces are used in Java to achieve abstraction. By using the implements keyword, a java class can implement an interface. In general terms, an interface can be defined as a container that stores the signatures of the methods to be implemented in the code segment.

Q6.What is the lambda expression of Java 8?

Ans - Lambda expression is a Java programming language feature introduced in Java 8 that provides functional programming constructs to the Java programming language, which simplifies the Java code in certain cases such as with Java anonymous inner classes.

Q7.Can you pass lambda expressions to a method? When?

Ans -If we need to pass a lambda expression as an argument, the type of parameter receiving the lambda expression argument must be of a functional interface type.

Q8.What is the functional interface in Java 8?

Ans - An interface is called a functional interface if it has a single abstract method irrespective of the number of default or static methods. Functional Interface are use for lamda expression. Runnable , Callable , Comparable , Comparator are few examples of Functional Interface.

Q9.What is the benefit of lambda expressions in Java 8?

Ans - 1.Lambda expressions improve code readability and do not require interpretation.
2.Lambdas allow you to write concise code.

3. It encourages the use of functional programming.

4. It simplifies variable scope and encourages code reusability.

Q10.Is it mandatory for a lambda expression to have parameters?

Ans - No need to declare the type of a parameter. The compiler can inference the same from the value of the parameter.