Java

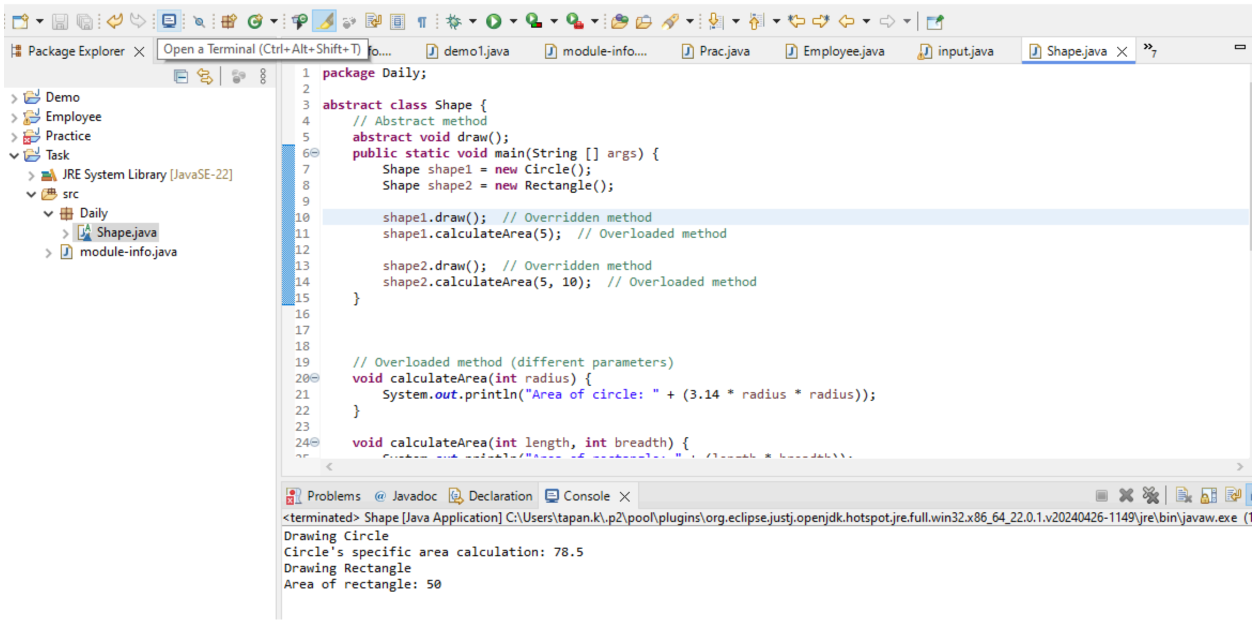
1. Implement Abstract class with overloading and overriding?

Answer:-

```
package Daily;
public abstract class Shape {


  // Abstract method
  abstract void draw();
  public static void main(String[] args) {
      Shape shape1 = new Circle();
      Shape shape2 = new Rectangle();

      shape1.draw();  // Overridden method
      shape1.calculateArea(5);  // Overloaded method
      shape2.draw();  // Overridden method
      shape2.calculateArea(5, 10);  // Overloaded method
  }
  // Overloaded method (different parameters)
  void calculateArea(int radius) {
      System.out.println("Area of circle: " + (3.14 * radius * radius));
  }
  void calculateArea(int length, int breadth) {
      System.out.println("Area of rectangle: " + (length * breadth));
  }
}
class Circle extends Shape {
  // Overriding abstract method
  @Override
  void draw() {
      System.out.println("Drawing Circle");
  }

  // Overriding calculateArea (Optional)
  @Override
  void calculateArea(int radius) {
      System.out.println("Circle's specific area calculation: " + (3.14 * radius * radius));
  }
}
class Rectangle extends Shape {
  // Overriding abstract method
  @Override
  void draw() {
      System.out.println("Drawing Rectangle");
```

```
    }
}
```

Output:-



```
package Daily;

abstract class Shape {
    // Abstract method
    abstract void draw();
    public static void main(String [] args) {
        Shape shape1 = new Circle();
        Shape shape2 = new Rectangle();

        shape1.draw();  // Overridden method
        shape1.calculateArea(5);  // Overloaded method

        shape2.draw();  // Overridden method
        shape2.calculateArea(5, 10);  // Overloaded method
    }



    // Overloaded method (different parameters)
    void calculateArea(int radius) {
        System.out.println("Area of circle: " + (3.14 * radius * radius));
    }

    void calculateArea(int length, int breadth) {
```

```
<terminated> Shape [Java Application] C:\Users\tapan.k\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_22.0.1.v20240426-1149\jre\bin\javaw.exe
Drawing Circle
Circle's specific area calculation: 78.5
Drawing Rectangle
Area of rectangle: 50
```

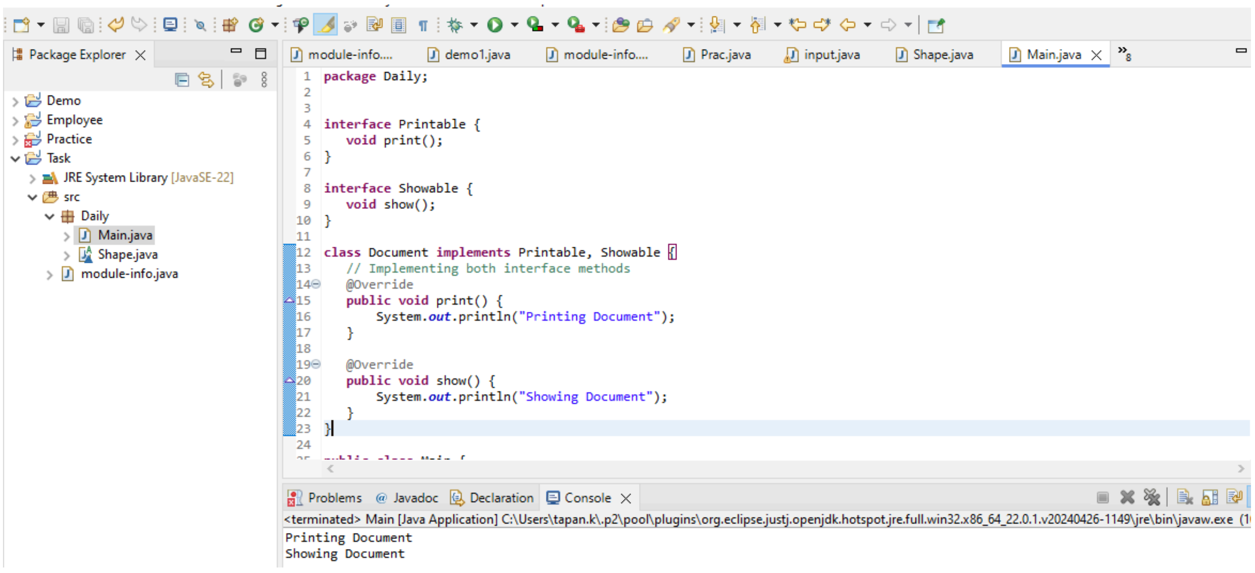
2. Implement Multiple inheritance with Interface?

Answer:-

```
package Daily;
interface Printable {
  void print();
}
interface Showable {
  void show();
}
class Document implements Printable, Showable {
  // Implementing both interface methods
  @Override
  public void print() {
      System.out.println("Printing Document");
  }
  @Override
  public void show() {
      System.out.println("Showing Document");
  }
}
```

```java
public class Main {
  public static void main(String[] args) {
      Document doc = new Document();
      doc.print();
      doc.show();
  }
}
```

Output:-



```java
package Daily;


interface Printable {
   void print();
}

interface Showable {
   void show();
}

class Document implements Printable, Showable {
   // Implementing both interface methods
   @Override
   public void print() {
       System.out.println("Printing Document");
   }

   @Override
   public void show() {
       System.out.println("Showing Document");
   }
}
```

```
<terminated> Main [Java Application] C:\Users\tapan.k\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_22.0.1.v20240426-1149\jre\bin\javaw.exe  (1
Printing Document
Showing Document
```

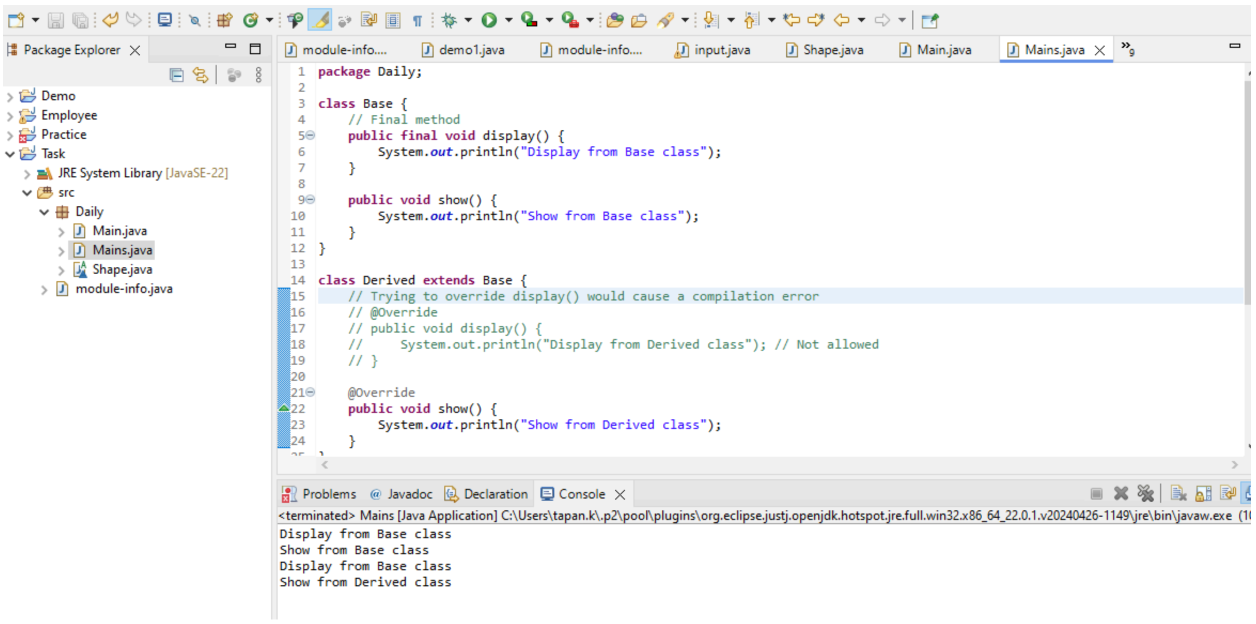
3. Show final methods in the class that can't be overridden?

Answer:-

```java
package Daily;
class Base {
   // Final method
   public final void display() {
       System.out.println("Display from Base class");
   }

   public void show() {
       System.out.println("Show from Base class");
   }
}
class Derived extends Base {
   // Trying to override display() would cause a compilation error
   // @Override
```

```java
    // public void display() {
    //     System.out.println("Display from Derived class"); // Not allowed
    // }
    @Override
    public void show() {
        System.out.println("Show from Derived class");
    }
}
public class Mains {
    public static void main(String[] args) {
        Base base = new Base();
        base.display();  // Final method
        base.show();     // Non-final method
        Derived derived = new Derived();
        derived.display();  // Final method from Base class
        derived.show();     // Overridden method from Derived class
    }
}
```

Output:-



```java
package Daily;

class Base {
    // Final method
    public final void display() {
        System.out.println("Display from Base class");
    }

    public void show() {
        System.out.println("Show from Base class");
    }
}

class Derived extends Base {
    // Trying to override display() would cause a compilation error
    // @Override
    // public void display() {
    //     System.out.println("Display from Derived class"); // Not allowed
    // }

    @Override
    public void show() {
        System.out.println("Show from Derived class");
    }
```

```
<terminated> Mains [Java Application] C:\Users\tapan.k\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_22.0.1.v20240426-1149\jre\bin\javaw.exe
Display from Base class
Show from Base class
Display from Base class
Show from Derived class
```