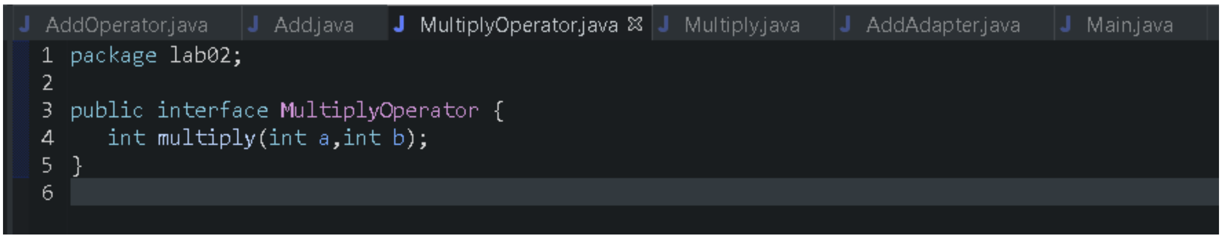
Lab Session 02

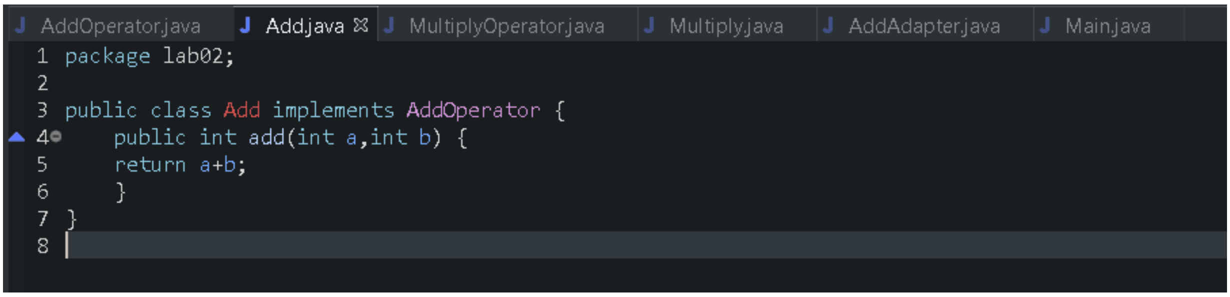
Practice Adapter Design Pattern to relate different interfaces in a software system

Exercise:

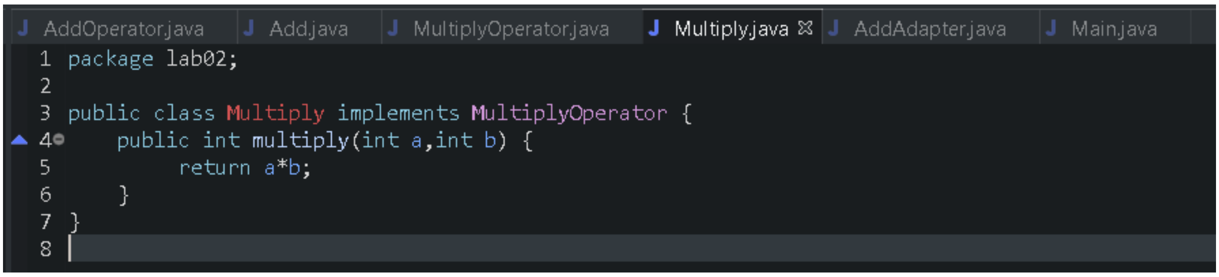
1.Develop a simple calculator following the Adapter Design Pattern.



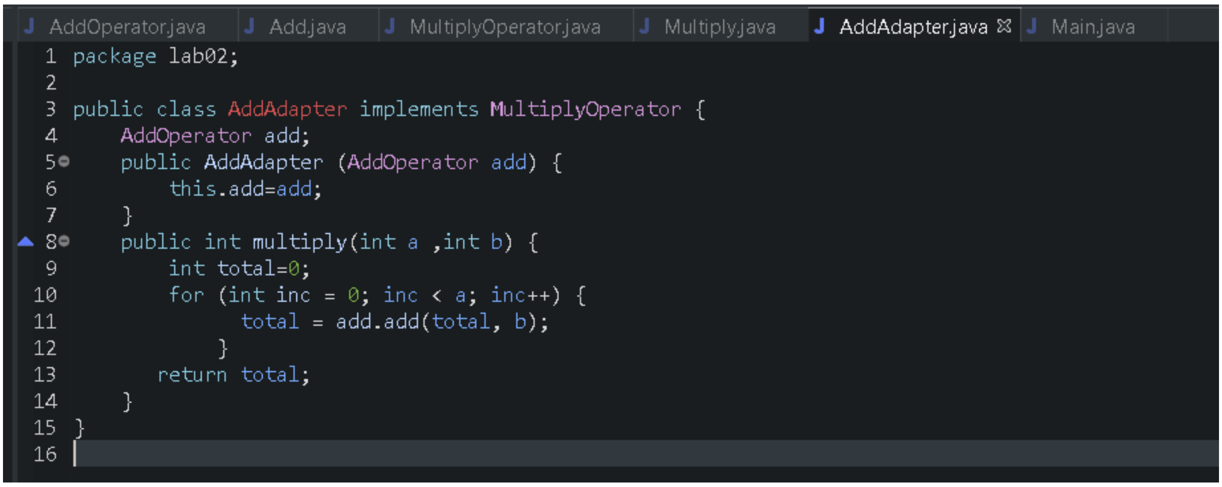
```
package lab02;

public interface MultiplyOperator {
   int multiply(int a,int b);
}
```



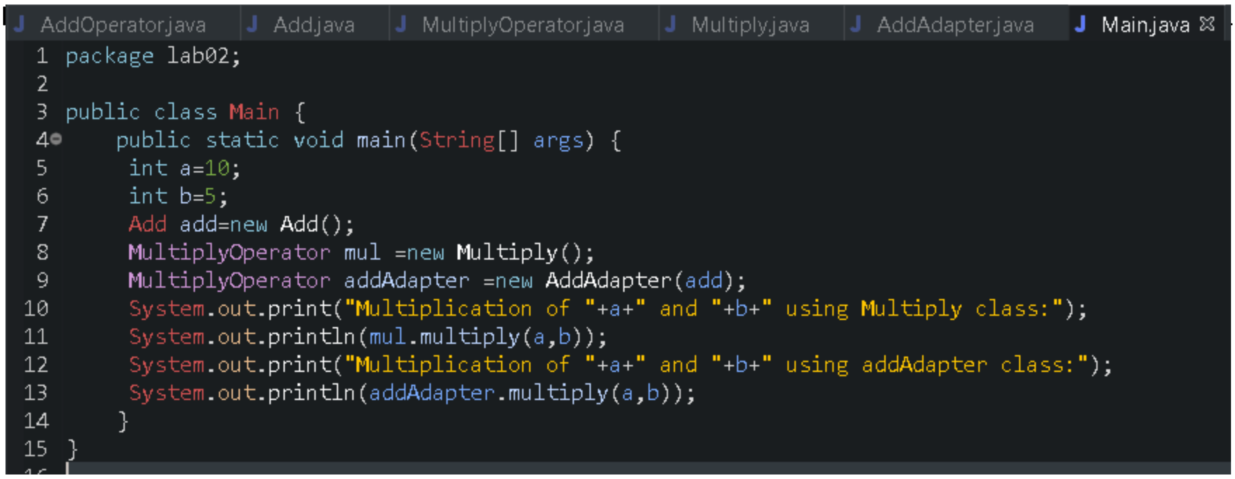
```
package lab02;

public class Add implements AddOperator {
    public int add(int a,int b) {
    return a+b;
    }
}
```



```
package lab02;

public class Multiply implements MultiplyOperator {
    public int multiply(int a,int b) {
        return a*b;
    }
}
```

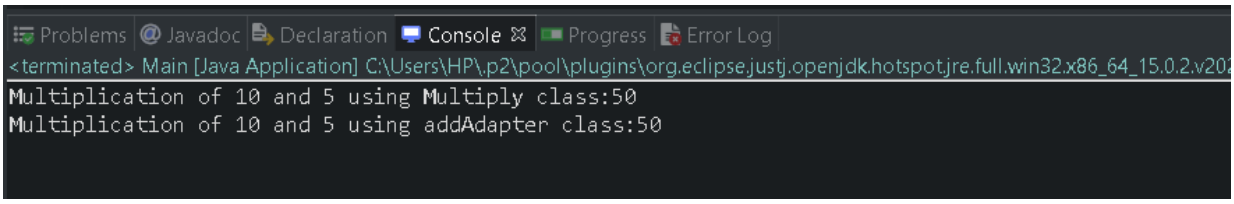


```
package lab02;

public class AddAdapter implements MultiplyOperator {
    AddOperator add;
    public AddAdapter (AddOperator add) {
        this.add=add;
    }
    public int multiply(int a ,int b) {
        int total=0;
        for (int inc = 0; inc < a; inc++) {
              total = add.add(total, b);
           }
       return total;
    }
}
```



```java
package lab02;

public class Main {
    public static void main(String[] args) {
     int a=10;
     int b=5;
     Add add=new Add();
     MultiplyOperator mul =new Multiply();
     MultiplyOperator addAdapter =new AddAdapter(add);
     System.out.print("Multiplication of "+a+" and "+b+" using Multiply class:");
     System.out.println(mul.multiply(a,b));
     System.out.print("Multiplication of "+a+" and "+b+" using addAdapter class:");
     System.out.println(addAdapter.multiply(a,b));
    }
}
```

Output:



```
<terminated> Main [Java Application] C:\Users\HP\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.2.v202
Multiplication of 10 and 5 using Multiply class:50
Multiplication of 10 and 5 using addAdapter class:50
```

2. Provide the code for the following diagram:

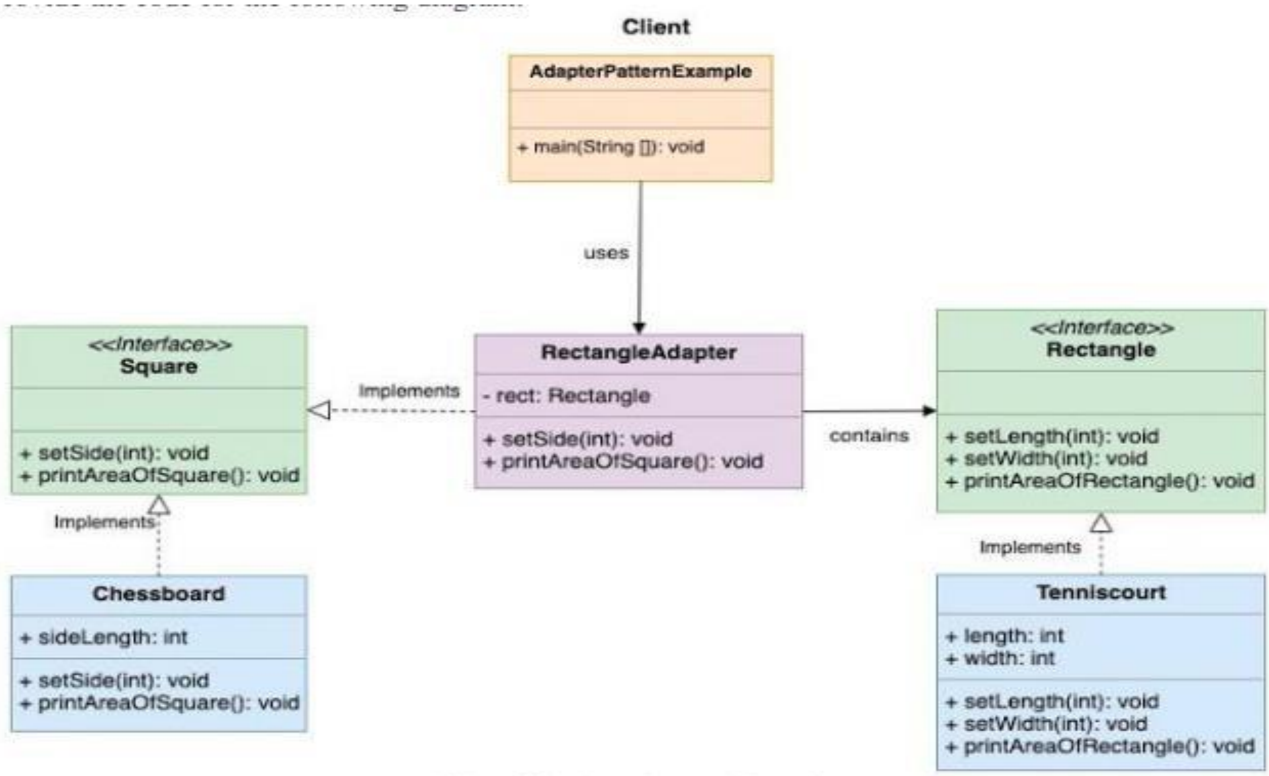
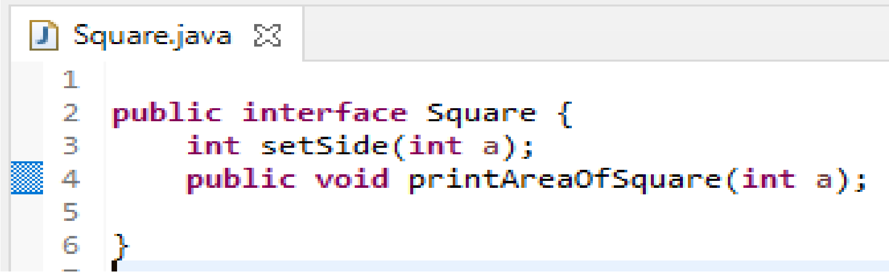
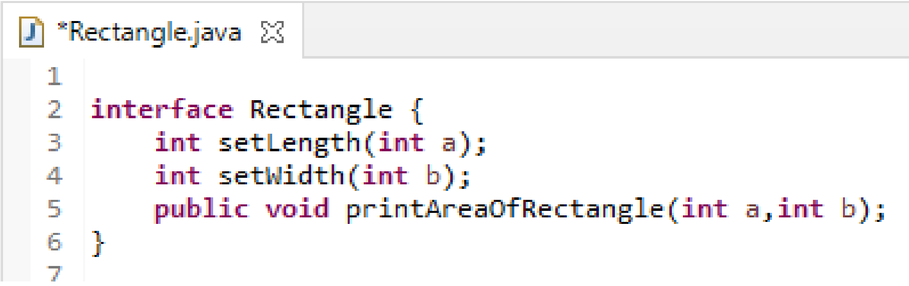


Figure 2.2 Adapter Pattern Example

Step 1: Create Square.java and Rectangle.java interfaces

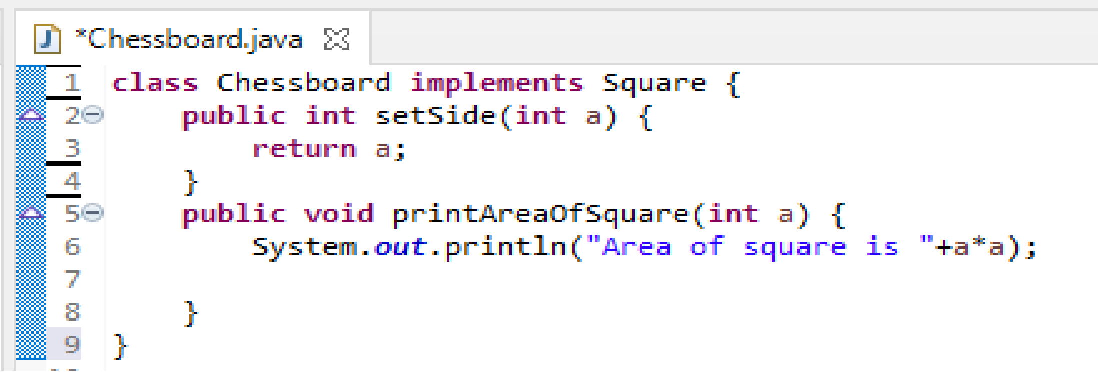


```java

public interface Square {
	int setSide(int a);
	public void printAreaOfSquare(int a);

}
```

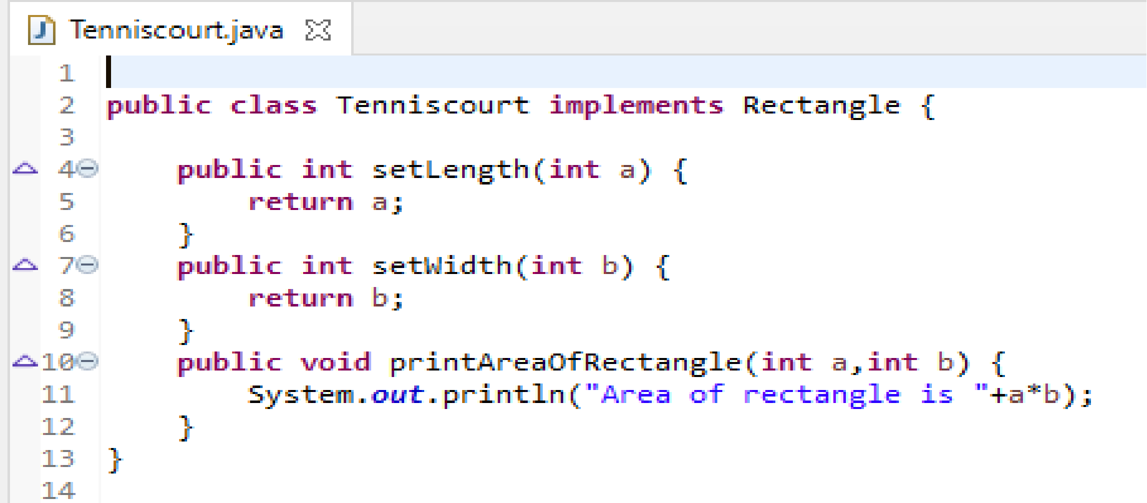


```java

interface Rectangle {
	int setLength(int a);
	int setWidth(int b);
	public void printAreaOfRectangle(int a,int b);
}
```

Step 2: Create concrete classes implementing the interfaces

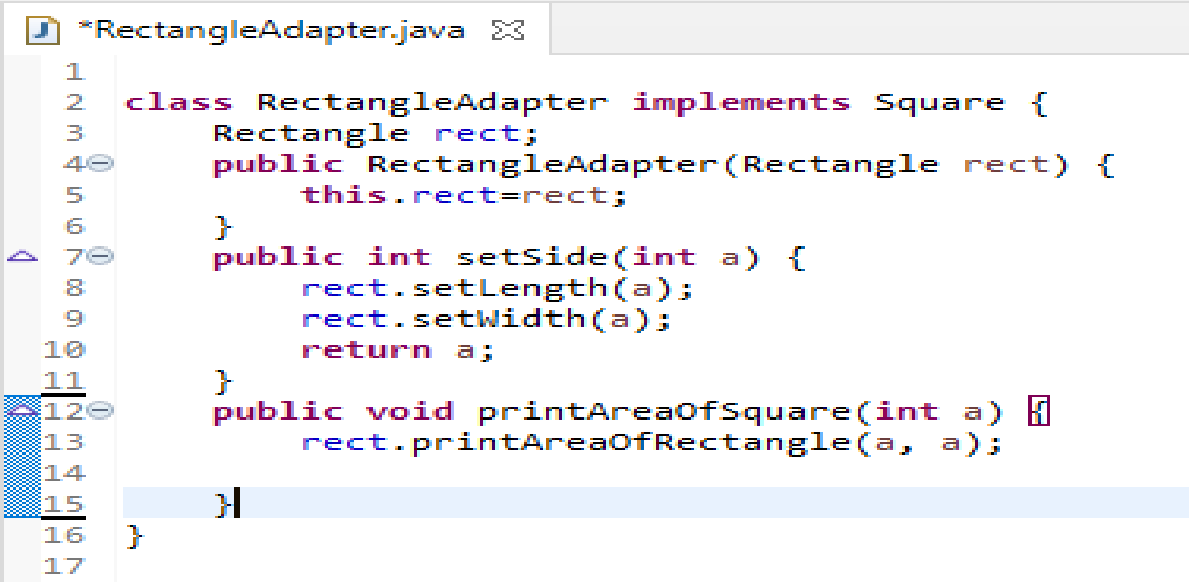


```java
class Chessboard implements Square {
    public int setSide(int a) {
        return a;
    }
    public void printAreaOfSquare(int a) {
        System.out.println("Area of square is "+a*a);

    }
}
```



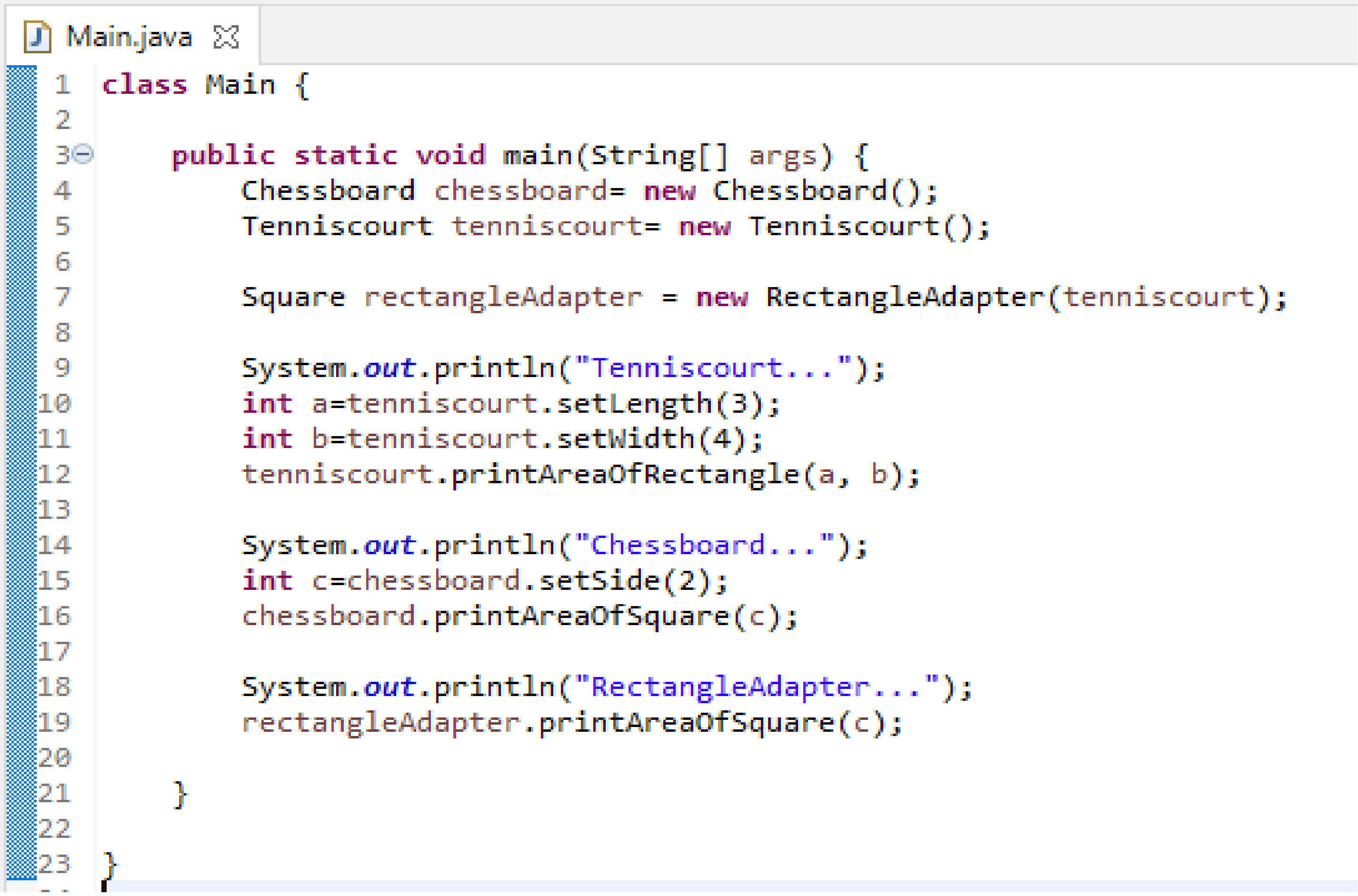
```java

public class Tenniscourt implements Rectangle {

    public int setLength(int a) {
        return a;
    }
    public int setWidth(int b) {
        return b;
    }
    public void printAreaOfRectangle(int a,int b) {
        System.out.println("Area of rectangle is "+a*b);
    }
}

```

Step 3: Create the adapter class implementing the adaptee interface



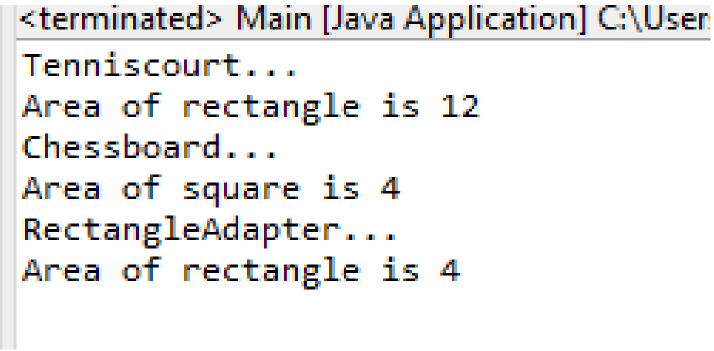
```java

class RectangleAdapter implements Square {
    Rectangle rect;
    public RectangleAdapter(Rectangle rect) {
        this.rect=rect;
    }
    public int setSide(int a) {
        rect.setLength(a);
        rect.setWidth(a);
        return a;
    }
    public void printAreaOfSquare(int a) {
        rect.printAreaOfRectangle(a, a);

    }
}

```

Step 4: Create the main class demonstrating the adapter design pattern



```java
class Main {

    public static void main(String[] args) {
        Chessboard chessboard= new Chessboard();
        Tenniscourt tenniscourt= new Tenniscourt();

        Square rectangleAdapter = new RectangleAdapter(tenniscourt);

        System.out.println("Tenniscourt...");
        int a=tenniscourt.setLength(3);
        int b=tenniscourt.setWidth(4);
        tenniscourt.printAreaOfRectangle(a, b);

        System.out.println("Chessboard...");
        int c=chessboard.setSide(2);
        chessboard.printAreaOfSquare(c);

        System.out.println("RectangleAdapter...");
        rectangleAdapter.printAreaOfSquare(c);

    }

}
```

Output:



```
Tenniscourt...
Area of rectangle is 12
Chessboard...
Area of square is 4
RectangleAdapter...
Area of rectangle is 4
```