**SOLID Principle:**

**1)Open/Closed Principle (OCP)**

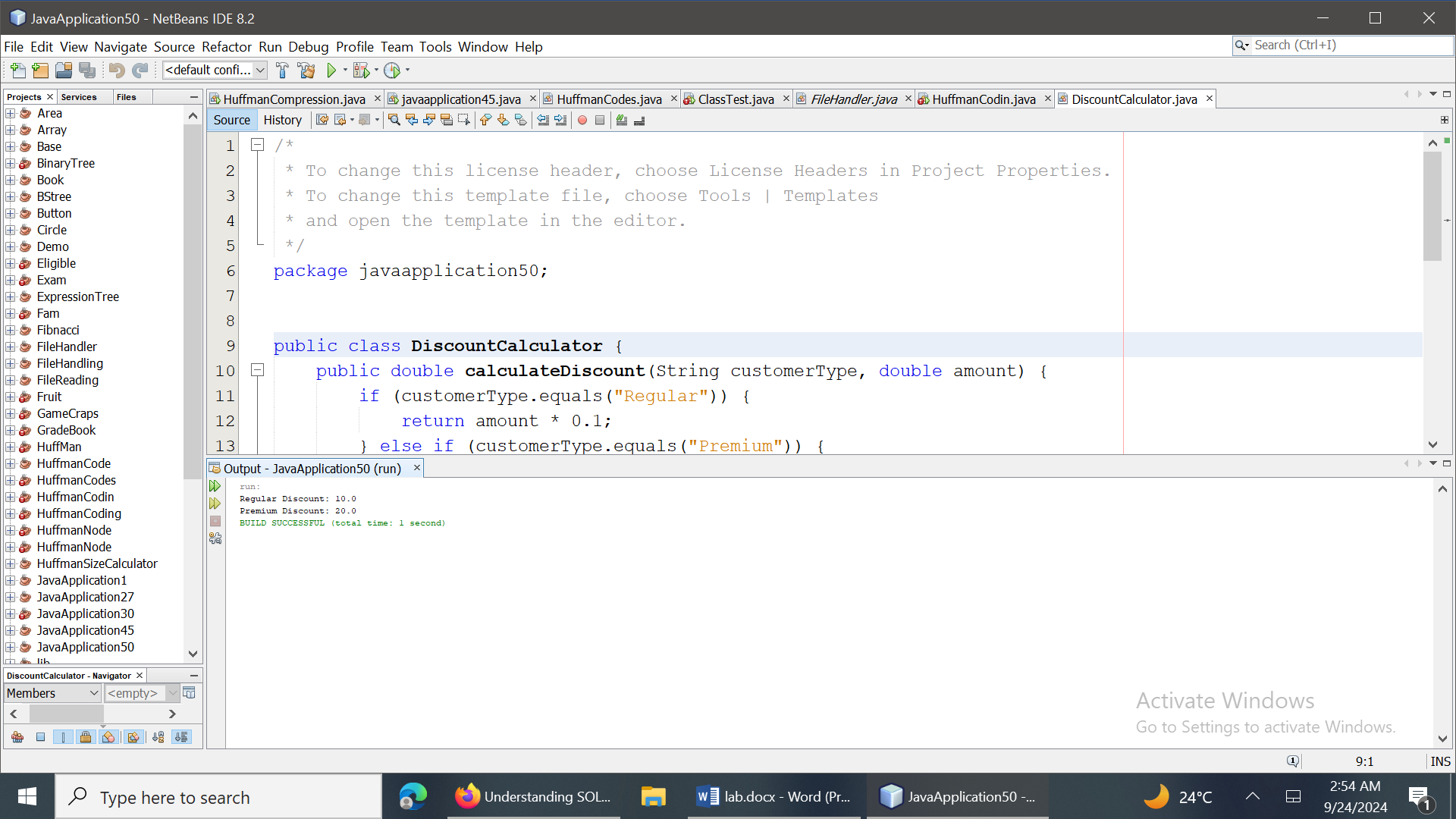
**Definition:** Software entities should be open for extension but closed for modification.

**Before OCP:**

Suppose we have a class that calculates discounts for different types of customers.

public class DiscountCalculator {  
 public double calculateDiscount(String customerType, double amount) {  
 if (customerType.equals("Regular")) {  
 return amount \* 0.1;  
 } else if (customerType.equals("Premium")) {  
 return amount \* 0.2;  
 }  
 return 0;  
 }  
  
 public static void main(String[] args) {  
 DiscountCalculator calculator = new DiscountCalculator();  
 double regularDiscount = calculator.calculateDiscount("Regular", 100);  
 double premiumDiscount = calculator.calculateDiscount("Premium", 100);  
  
 System.out.println("Regular Discount: " + regularDiscount);  
 System.out.println("Premium Discount: " + premiumDiscount);  
 }  
}

**OUTPUT:**

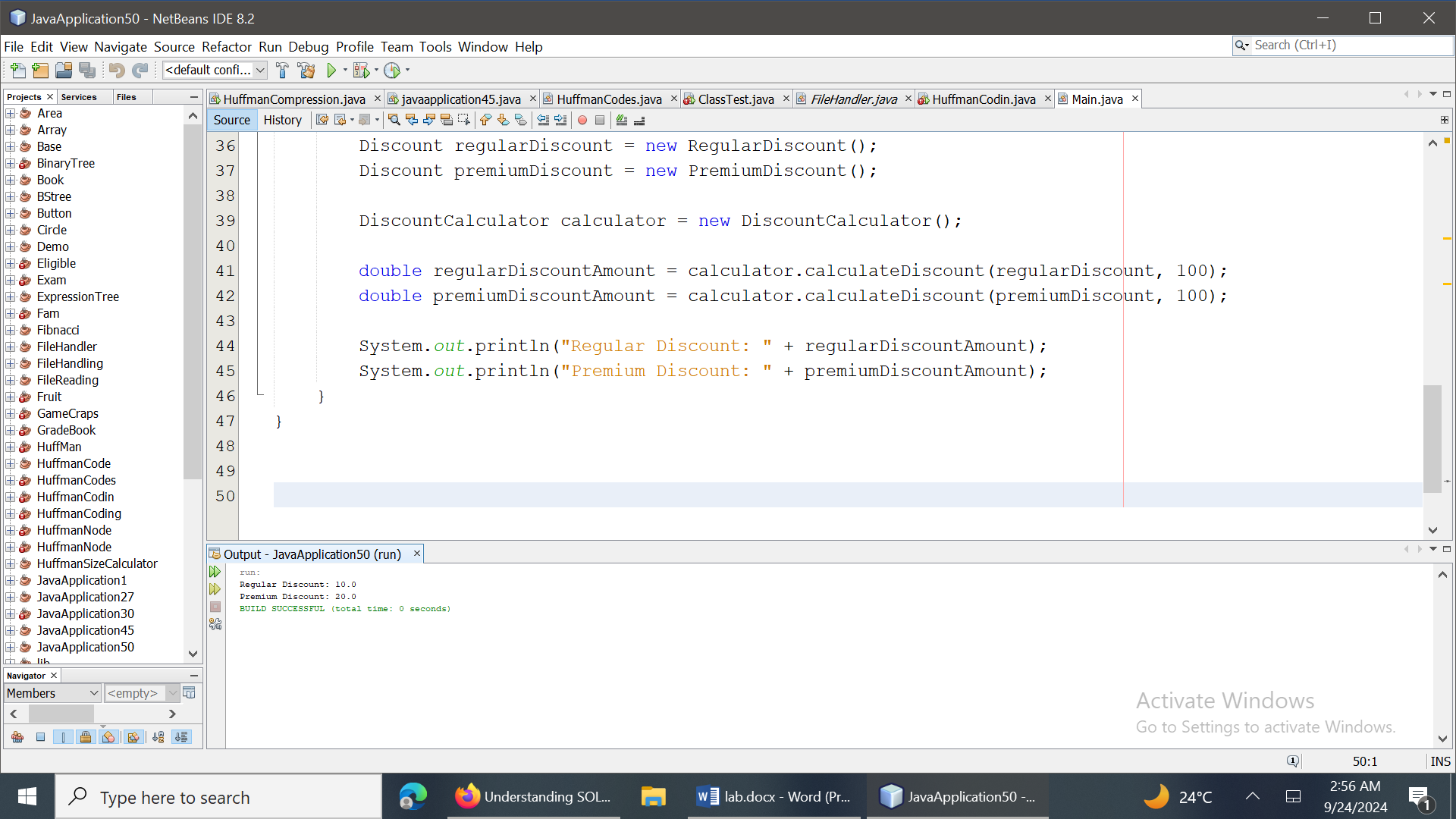
****

**Explanation:** The DiscountCalculator class violates the Open/Closed Principle because it needs to be modified whenever a new type of discount is added.

**After OCP:**

To adhere to OCP, we can use polymorphism to extend the behavior without modifying the existing code.

public abstract class Discount {  
 public abstract double calculate(double amount);  
}  
  
// RegularDiscount.java  
public class RegularDiscount extends Discount {  
 public double calculate(double amount) {  
 return amount \* 0.1;  
 }  
}  
  
// PremiumDiscount.java  
public class PremiumDiscount extends Discount {  
 public double calculate(double amount) {  
 return amount \* 0.2;  
 }  
}  
  
// DiscountCalculator.java  
public class DiscountCalculator {  
 public double calculateDiscount(Discount discount, double amount) {  
 return discount.calculate(amount);  
 }  
}  
  
// Main.java  
public class Main {  
 public static void main(String[] args) {  
 Discount regularDiscount = new RegularDiscount();  
 Discount premiumDiscount = new PremiumDiscount();  
  
 DiscountCalculator calculator = new DiscountCalculator();  
  
 double regularDiscountAmount = calculator.calculateDiscount(regularDiscount, 100);  
 double premiumDiscountAmount = calculator.calculateDiscount(premiumDiscount, 100);  
  
 System.out.println("Regular Discount: " + regularDiscountAmount);  
 System.out.println("Premium Discount: " + premiumDiscountAmount);  
 }  
}

**Output: **

**Explanation:** The DiscountCalculator class now depends on the Discount interface, allowing new discount types to be added without modifying the existing code. This adheres to the Open/Closed Principle.