SCD-Lab

Lab#11

Name: Anas-Altaf

Roll.No: 22F-3639

Java-Junit Tests:

OrderProcessor.java

```java
package org.scblab;

public class OrderProcessor {
   public double processOrder(double price, int quantity, boolean
           isMember, boolean isHoliday) {

       double total = 0.0;
       if (price <= 0 || quantity <= 0) {
           return -1; // Invalid input
       }
       double discount = 0.0;
       if (isMember) {
           discount = price * 0.1;
       }
       if (isHoliday) {
           discount += price * 0.05;
       }
       if (checkInventory(quantity)) {
           total = (price - discount) * quantity;
       } else {
           return -2; // Out of stock
       }
       if (total > 500) {
           total *= 0.9; // Apply bulk discount
       }
       return total;
   }

   private boolean checkInventory(int quantity) {
// Assume we have limited stock
       return quantity <= 100;
   }
}
```

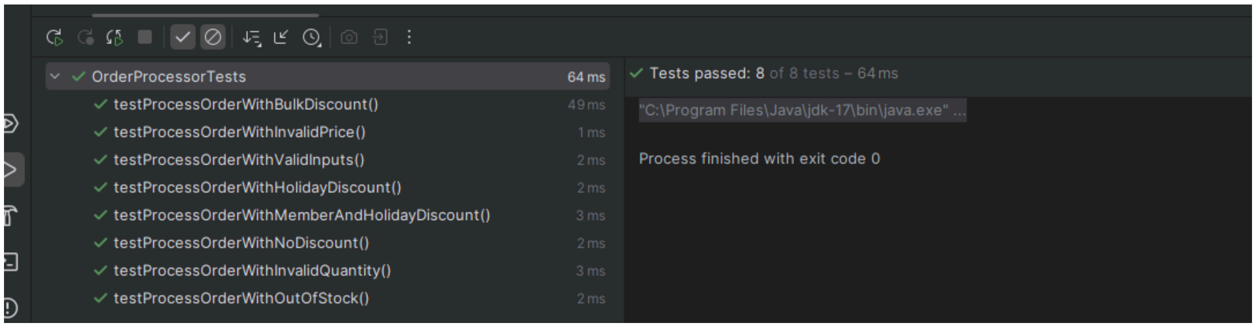
T-1:

```
import jdk.jfr.Name;
import org.junit.jupiter.api.Test;
import org.scblab.OrderProcessor;

import static org.junit.jupiter.api.Assertions.assertEquals;

public class OrderProcessorTests {

  @Test
  @Name("Test processOrder with valid inputs")
  public void testProcessOrderWithValidInputs() {
     OrderProcessor orderProcessor = new OrderProcessor();
     double price = 100.0;
     int quantity = 10;
     boolean isMember = true;
     boolean isHoliday = false;
     double expected = 810.0;
     double actual = orderProcessor.processOrder(price, quantity, isMember, isHoliday);
     assertEquals(expected, actual);
  }

  @Test
  @Name("Test processOrder with holiday discount")
  public void testProcessOrderWithHolidayDiscount() {
     OrderProcessor orderProcessor = new OrderProcessor();
     double price = 100.0;
     int quantity = 10;
     boolean isMember = false;
     boolean isHoliday = true;
     double expected = 855.0;
     double actual = orderProcessor.processOrder(price, quantity, isMember, isHoliday);
     assertEquals(expected, actual);
  }

  @Test
  @Name("Test processOrder with member discount")
  public void testProcessOrderWithMemberAndHolidayDiscount() {
     OrderProcessor orderProcessor = new OrderProcessor();
     double price = 100.0;
     int quantity = 10;
     boolean isMember = true;
     boolean isHoliday = true;
     double expected = 765.0;
```

```java
    double actual = orderProcessor.processOrder(price, quantity, isMember, isHoliday);
    assertEquals(expected, actual);
}

@Test
@Name("Test processOrder with invalid price")
public void testProcessOrderWithInvalidPrice() {
    OrderProcessor orderProcessor = new OrderProcessor();
    double price = -100.0;
    int quantity = 10;
    boolean isMember = true;
    boolean isHoliday = false;
    double expected = -1.0;
    double actual = orderProcessor.processOrder(price, quantity, isMember, isHoliday);
    assertEquals(expected, actual);
}

@Test
@Name("Test processOrder with invalid quantity")
public void testProcessOrderWithInvalidQuantity() {
    OrderProcessor orderProcessor = new OrderProcessor();
    double price = 100.0;
    int quantity = -10;
    boolean isMember = true;
    boolean isHoliday = false;
    double expected = -1.0;
    double actual = orderProcessor.processOrder(price, quantity, isMember, isHoliday);
    assertEquals(expected, actual);
}

@Test
@Name("Test processOrder with out of stock")
public void testProcessOrderWithOutOfStock() {
    OrderProcessor orderProcessor = new OrderProcessor();
    double price = 100.0;
    int quantity = 200;
    boolean isMember = true;
    boolean isHoliday = false;
    double expected = -2.0;
    double actual = orderProcessor.processOrder(price, quantity, isMember, isHoliday);
    assertEquals(expected, actual);
}

@Test
@Name("Test processOrder with bulk discount")
public void testProcessOrderWithBulkDiscount() {
    OrderProcessor orderProcessor = new OrderProcessor();
    double price = 100.0;
    int quantity = 100;
```

```
        boolean isMember = true;
        boolean isHoliday = false;
        double expected = 8100.0;
        double actual = orderProcessor.processOrder(price, quantity, isMember, isHoliday);
        assertEquals(expected, actual);
    }

    @Test
    @Name("Test processOrder with no discount")
    public void testProcessOrderWithNoDiscount() {
        OrderProcessor orderProcessor = new OrderProcessor();
        double price = 100.0;
        int quantity = 10;
        boolean isMember = false;
        boolean isHoliday = false;
        double expected = 900.0;
        double actual = orderProcessor.processOrder(price, quantity, isMember, isHoliday);
        assertEquals(expected, actual);
    }

}
```

Output:



```
OrderProcessorTests                                  64 ms
  ✓ testProcessOrderWithBulkDiscount()               49 ms
  ✓ testProcessOrderWithInvalidPrice()                1 ms
  ✓ testProcessOrderWithValidInputs()                 2 ms
  ✓ testProcessOrderWithHolidayDiscount()             2 ms
  ✓ testProcessOrderWithMemberAndHolidayDiscount()    3 ms
  ✓ testProcessOrderWithNoDiscount()                  2 ms
  ✓ testProcessOrderWithInvalidQuantity()             3 ms
  ✓ testProcessOrderWithOutOfStock()                  2 ms

✓ Tests passed: 8 of 8 tests – 64 ms

"C:\Program Files\Java\jdk-17\bin\java.exe" ...

Process finished with exit code 0
```

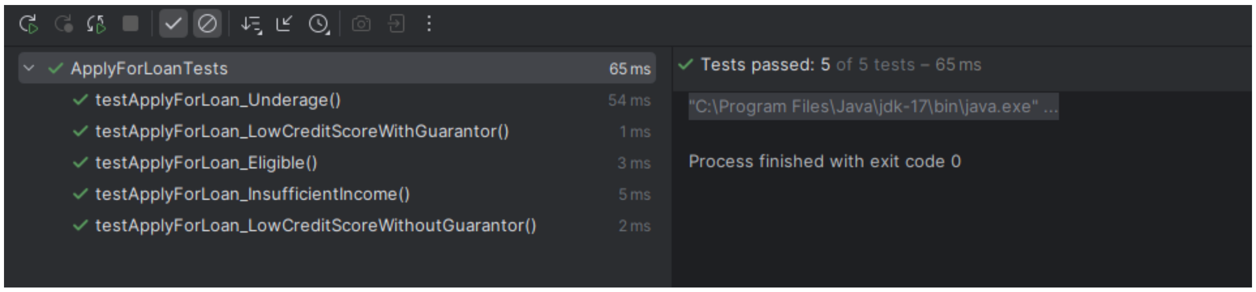
LoadManagementSystem.java

T-2:

Apply For Loan Test:

```
import org.junit.jupiter.api.Test;
import org.scblab.LoanManagementSystem;

import static org.junit.jupiter.api.Assertions.assertEquals;

public class ApplyForLoanTests {
    @Test
    public void testApplyForLoan_Underage() {
        LoanManagementSystem lms = new LoanManagementSystem();
        String result = lms.applyForLoan(17, 30000, 650, false);
        assertEquals("Ineligible: Underage", result);
    }

    @Test
    public void testApplyForLoan_InsufficientIncome() {
        LoanManagementSystem lms = new LoanManagementSystem();
        String result = lms.applyForLoan(25, 15000, 700, false);
        assertEquals("Ineligible: Insufficient income", result);
    }

    @Test
    public void testApplyForLoan_LowCreditScoreWithoutGuarantor() {
        LoanManagementSystem lms = new LoanManagementSystem();
        String result = lms.applyForLoan(30, 30000, 500, false);
        assertEquals("Ineligible: Low credit score", result);
    }

    @Test
    public void testApplyForLoan_LowCreditScoreWithGuarantor() {
        LoanManagementSystem lms = new LoanManagementSystem();
        String result = lms.applyForLoan(30, 30000, 500, true);
        assertEquals("Eligible with guarantor", result);
    }

    @Test
    public void testApplyForLoan_Eligible() {
        LoanManagementSystem lms = new LoanManagementSystem();
        String result = lms.applyForLoan(30, 30000, 700, false);
        assertEquals("Eligible", result);
    }
}
```

Output:

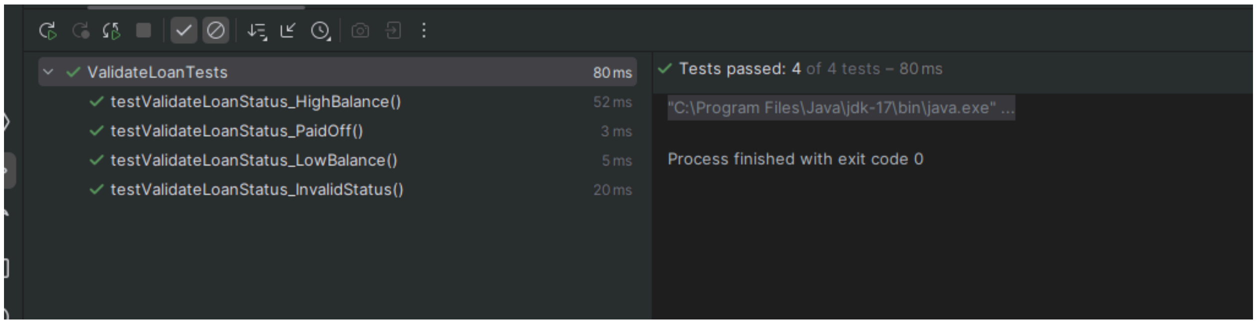


```
✓ ApplyForLoanTests                                          65 ms
    ✓ testApplyForLoan_Underage()                            54 ms
    ✓ testApplyForLoan_LowCreditScoreWithGuarantor()          1 ms
    ✓ testApplyForLoan_Eligible()                             3 ms
    ✓ testApplyForLoan_InsufficientIncome()                   5 ms
    ✓ testApplyForLoan_LowCreditScoreWithoutGuarantor()       2 ms

✓ Tests passed: 5 of 5 tests – 65 ms

"C:\Program Files\Java\jdk-17\bin\java.exe" ...

Process finished with exit code 0
```

Validate Loan Tests:

```java
import org.junit.jupiter.api.Test;
import org.scblab.LoanManagementSystem;

import static org.junit.jupiter.api.Assertions.assertEquals;

public class ValidateLoanTests {

  @Test
  public void testValidateLoanStatus_PaidOff() {
      LoanManagementSystem lms = new LoanManagementSystem();
      String result = lms.validateLoanStatus(0);
      assertEquals("Loan status: Paid off", result);
  }

  @Test
  public void testValidateLoanStatus_LowBalance() {
      LoanManagementSystem lms = new LoanManagementSystem();
      String result = lms.validateLoanStatus(3000);
      assertEquals("Loan status: Low balance", result);
  }

  @Test
  public void testValidateLoanStatus_HighBalance() {
      LoanManagementSystem lms = new LoanManagementSystem();
      String result = lms.validateLoanStatus(6000);
      assertEquals("Loan status: High balance", result);
  }

  @Test
  public void testValidateLoanStatus_InvalidStatus() {
      LoanManagementSystem lms = new LoanManagementSystem();
      String result = lms.validateLoanStatus(-100);
      assertEquals("Invalid loan status", result);
  }
}
```

Output:



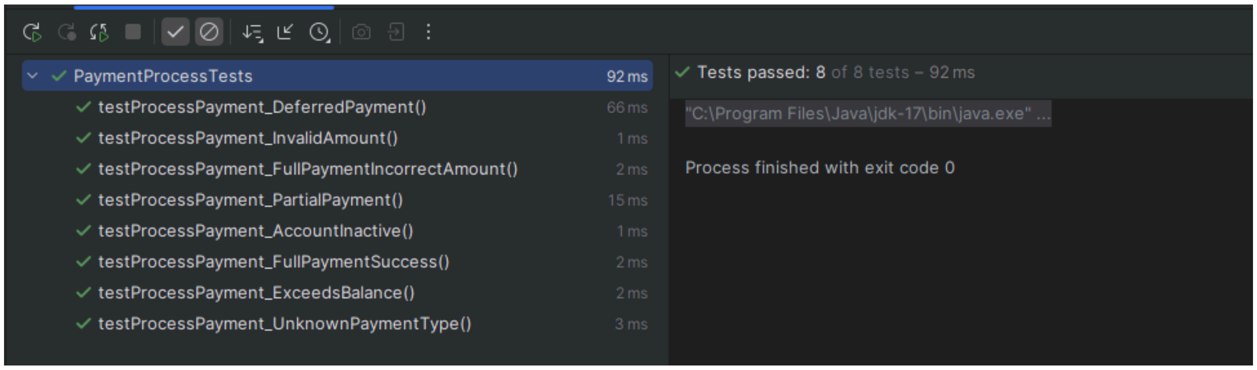
```
ValidateLoanTests                              80 ms
  testValidateLoanStatus_HighBalance()         52 ms
  testValidateLoanStatus_PaidOff()              3 ms
  testValidateLoanStatus_LowBalance()           5 ms
  testValidateLoanStatus_InvalidStatus()       20 ms

Tests passed: 4 of 4 tests – 80 ms

"C:\Program Files\Java\jdk-17\bin\java.exe" ...

Process finished with exit code 0
```

Payment Processing Test:

```
import org.junit.jupiter.api.Test;
import org.scblab.LoanManagementSystem;

import static org.junit.jupiter.api.Assertions.assertEquals;

public class PaymentProcessTests {

  @Test
  public void testProcessPayment_AccountInactive() {
     LoanManagementSystem lms = new LoanManagementSystem();
     String result = lms.processPayment(1000, 500, "FULL", false);
     assertEquals("Payment failed: Account inactive", result);
  }

  @Test
  public void testProcessPayment_InvalidAmount() {
     LoanManagementSystem lms = new LoanManagementSystem();
     String result = lms.processPayment(1000, -500, "FULL", true);
     assertEquals("Payment failed: Invalid amount", result);
  }

  @Test
  public void testProcessPayment_ExceedsBalance() {
     LoanManagementSystem lms = new LoanManagementSystem();
     String result = lms.processPayment(1000, 1500, "FULL", true);
     assertEquals("Payment failed: Exceeds balance", result);
  }

  @Test
  public void testProcessPayment_FullPaymentSuccess() {
     LoanManagementSystem lms = new LoanManagementSystem();
     String result = lms.processPayment(1000, 1000, "FULL", true);
     assertEquals("Payment successful: Loan fully paid", result);
  }
```

```
    @Test
    public void testProcessPayment_FullPaymentIncorrectAmount() {
        LoanManagementSystem lms = new LoanManagementSystem();
        String result = lms.processPayment(1000, 500, "FULL", true);
        assertEquals("Payment failed: Incorrect amount for full payment", result);
    }

    @Test
    public void testProcessPayment_PartialPayment() {
        LoanManagementSystem lms = new LoanManagementSystem();
        String result = lms.processPayment(1000, 500, "PARTIAL", true);
        assertEquals("Payment successful: Partial payment of 500.0", result);
    }

    @Test
    public void testProcessPayment_DeferredPayment() {
        LoanManagementSystem lms = new LoanManagementSystem();
        String result = lms.processPayment(1000, 500, "DEFERRED", true);
        assertEquals("Payment deferred", result);
    }

    @Test
    public void testProcessPayment_UnknownPaymentType() {
        LoanManagementSystem lms = new LoanManagementSystem();
        String result = lms.processPayment(1000, 500, "UNKNOWN", true);
        assertEquals("Payment failed: Unknown payment type", result);
    }
}
```

Output:



```
✓ PaymentProcessTests                                   92 ms
    ✓ testProcessPayment_DeferredPayment()              66 ms
    ✓ testProcessPayment_InvalidAmount()                 1 ms
    ✓ testProcessPayment_FullPaymentIncorrectAmount()    2 ms
    ✓ testProcessPayment_PartialPayment()               15 ms
    ✓ testProcessPayment_AccountInactive()               1 ms
    ✓ testProcessPayment_FullPaymentSuccess()            2 ms
    ✓ testProcessPayment_ExceedsBalance()                2 ms
    ✓ testProcessPayment_UnknownPaymentType()            3 ms

✓ Tests passed: 8 of 8 tests – 92 ms
"C:\Program Files\Java\jdk-17\bin\java.exe" ...

Process finished with exit code 0
```

Thanks