To verify the business rules implemented in your classes using **JUnit** (one of the most popular testing frameworks for Java), I’ll guide you through writing unit tests. Since you mentioned business rules in classes like PatientManagement, PrescriptionService, and TreatmentPlan, we will create **JUnit tests** for the following:

1. **Treatment Plan Creation**:
   * Verifying that the treatment plan is generated correctly for different types of patients (Inpatient vs. Outpatient).
2. **Prescription Generation**:
   * Verifying that the generatePrescription method in both OnlinePrescriptionService and PrintablePrescriptionService creates prescriptions in the correct format.
3. **Business Logic Validation**:
   * Verifying that age-based and treatment-type-based business logic works as expected.

We’ll write test cases to cover the following:

* Correct creation of treatment plans based on patient data.
* Validation of prescription formats generated in HTML and plain text.
* Handling edge cases like missing or invalid data.

**Step-by-Step JUnit Test Setup**

**1. Add JUnit Dependency**

If you're using Maven, add the following to your pom.xml:

xml

Copy code

<dependencies>

<!-- JUnit dependency -->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

</dependencies>

If you're not using Maven, ensure that your project includes the JUnit JAR (which you already have, according to your setup).

**2. Write the JUnit Tests**

We will write unit tests to verify that:

* Treatment plans are correctly created based on patient type.
* Prescriptions are generated in the expected format.

**Test for MedicationTreatmentPlan.java:**

java

Copy code

import static org.junit.Assert.assertEquals;

import com.algonquin.cst8288.fall24.assignment1.patient.Outpatient;

import com.algonquin.cst8288.fall24.assignment1.treatment.MedicationTreatmentPlan;

import org.junit.Before;

import org.junit.Test;

public class MedicationTreatmentPlanTest {

private MedicationTreatmentPlan medicationTreatmentPlan;

private Outpatient outpatient;

@Before

public void setUp() {

medicationTreatmentPlan = new MedicationTreatmentPlan();

outpatient = new Outpatient("002", "Jane Smith", "jane.smith@example.com", "9876543210", "1990-05-23", "2024-10-10");

outpatient.setLifeStage("ADULT");

outpatient.setPlannedTreatment("Infection");

}

@Test

public void testCreateTreatmentPlan() {

String expectedPlan = "Medication Treatment Plan:\n" +

"- Doses per day: 3\n" +

"- Duration: 14 days\n" +

"- Medication: Antibiotics";

String actualPlan = medicationTreatmentPlan.createTreatmentPlan(outpatient);

assertEquals(expectedPlan, actualPlan);

}

}

**Test for SurgeryTreatmentPlan.java:**

java

Copy code

import static org.junit.Assert.assertEquals;

import com.algonquin.cst8288.fall24.assignment1.patient.Inpatient;

import com.algonquin.cst8288.fall24.assignment1.treatment.SurgeryTreatmentPlan;

import org.junit.Before;

import org.junit.Test;

public class SurgeryTreatmentPlanTest {

private SurgeryTreatmentPlan surgeryTreatmentPlan;

private Inpatient inpatient;

@Before

public void setUp() {

surgeryTreatmentPlan = new SurgeryTreatmentPlan();

inpatient = new Inpatient("001", "John Doe", "john.doe@example.com", "1234567890", "1987-10-17", "Room 101");

inpatient.setAge(25); // Adult

}

@Test

public void testCreateTreatmentPlan() {

String expectedPlan = "Surgery Treatment Plan:\n" +

"- Fasting Requirement: 8 Hours Before Surgery\n" +

"- Follow-up Schedule: In a Month";

String actualPlan = surgeryTreatmentPlan.createTreatmentPlan(inpatient);

assertEquals(expectedPlan, actualPlan);

}

}

**Test for OnlinePrescriptionService.java:**

java

Copy code

import static org.junit.Assert.assertTrue;

import com.algonquin.cst8288.fall24.assignment1.patient.Patient;

import com.algonquin.cst8288.fall24.assignment1.prescription.OnlinePrescriptionService;

import com.algonquin.cst8288.fall24.assignment1.prescription.Prescription;

import com.algonquin.cst8288.fall24.assignment1.patient.Inpatient;

import org.junit.Before;

import org.junit.Test;

public class OnlinePrescriptionServiceTest {

private OnlinePrescriptionService onlinePrescriptionService;

private Patient patient;

@Before

public void setUp() {

onlinePrescriptionService = new OnlinePrescriptionService();

patient = new Inpatient("001", "John Doe", "john.doe@example.com", "1234567890", "1987-10-17", "Room 101");

patient.setPrescription(new Prescription("Surgery: Appendectomy"));

}

@Test

public void testGeneratePrescription() {

String prescription = onlinePrescriptionService.generatePrescription(patient);

assertTrue(prescription.contains("<h1>Prescription</h1>"));

assertTrue(prescription.contains("<p><strong>Patient Name:</strong> John Doe</p>"));

assertTrue(prescription.contains("<p><strong>Prescription Details:</strong> Surgery: Appendectomy</p>"));

}

}

**Test for PrintablePrescriptionService.java:**

java

Copy code

import static org.junit.Assert.assertTrue;

import com.algonquin.cst8288.fall24.assignment1.patient.Patient;

import com.algonquin.cst8288.fall24.assignment1.prescription.PrintablePrescriptionService;

import com.algonquin.cst8288.fall24.assignment1.prescription.Prescription;

import com.algonquin.cst8288.fall24.assignment1.patient.Outpatient;

import org.junit.Before;

import org.junit.Test;

public class PrintablePrescriptionServiceTest {

private PrintablePrescriptionService printablePrescriptionService;

private Patient patient;

@Before

public void setUp() {

printablePrescriptionService = new PrintablePrescriptionService();

patient = new Outpatient("002", "Jane Smith", "jane.smith@example.com", "9876543210", "1990-05-23", "2024-10-10");

patient.setPrescription(new Prescription("Medication: Amoxicillin"));

}

@Test

public void testGeneratePrescription() {

String prescription = printablePrescriptionService.generatePrescription(patient);

assertTrue(prescription.contains("Prescription"));

assertTrue(prescription.contains("Patient Name: Jane Smith"));

assertTrue(prescription.contains("Prescription Details: Medication: Amoxicillin"));

}

}

**Explanation of the Test Cases:**

1. **MedicationTreatmentPlanTest**:
   * Tests the creation of a medication treatment plan for an outpatient.
   * Asserts that the correct doses and duration are returned based on the patient's life stage and treatment type.
2. **SurgeryTreatmentPlanTest**:
   * Tests the creation of a surgery treatment plan for an inpatient.
   * Verifies that the fasting requirements and follow-up schedule are correctly returned based on the patient’s age.
3. **OnlinePrescriptionServiceTest**:
   * Tests the generation of an HTML-formatted prescription for a patient.
   * Verifies that the generated prescription contains key HTML tags and patient details.
4. **PrintablePrescriptionServiceTest**:
   * Tests the generation of a plain-text prescription for a patient.
   * Verifies that the prescription contains the correct patient details in plain text format.

**Step 3: Run Your Tests**

* You can run these tests using your IDE’s built-in JUnit test runner (like NetBeans or IntelliJ) or using Maven (mvn test).

This approach will verify that your business logic works as expected and that the key business rules (like life stage-based medication plans and age-based surgery plans) are functioning correctly.

Let me know if you need any further clarification or adjustments!