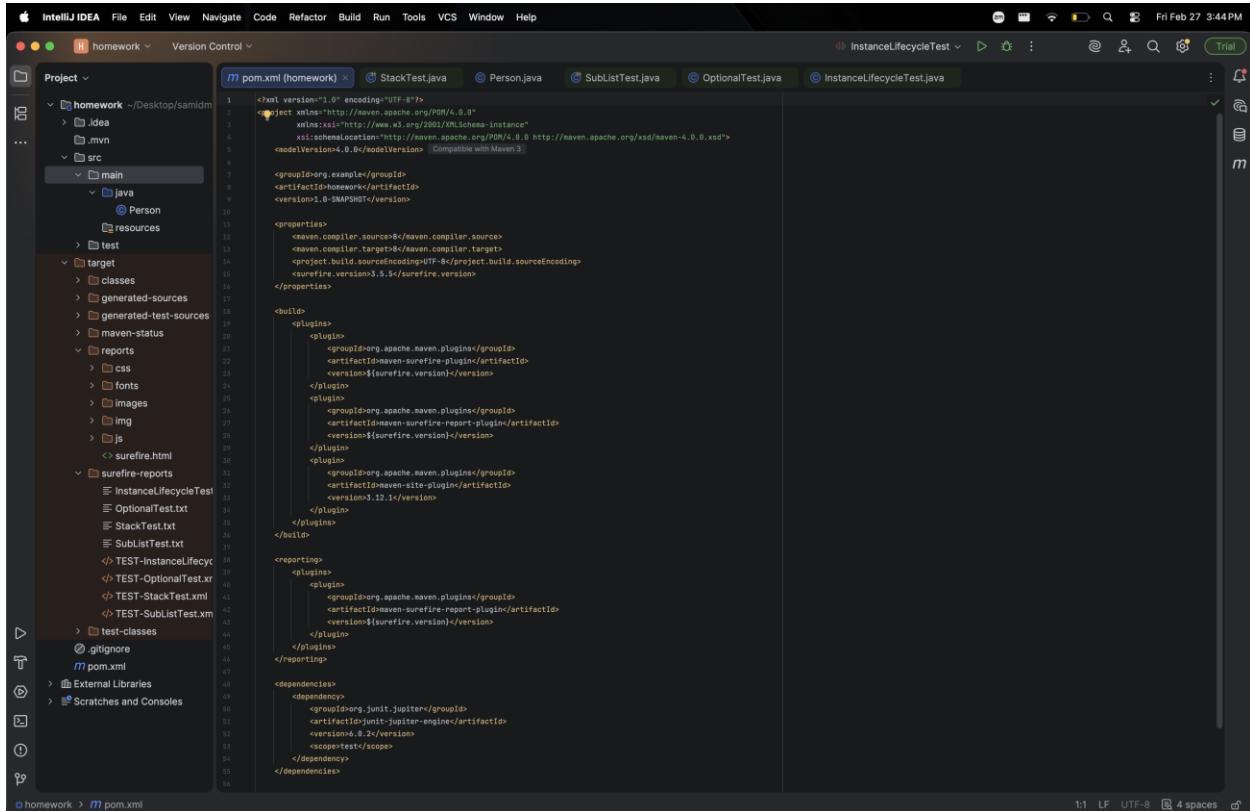


Week 2 – Day 4 Assignment (February 26th, 2026)

Test I – pom.xml



The screenshot shows the IntelliJ IDEA interface with the pom.xml file open in the central editor window. The code editor displays the XML configuration for a Maven project. The project structure on the left shows various test files like StackTest.java, Person.java, SubListTest.java, OptionalTest.java, and InstanceLifecycleTest.java. The pom.xml file itself contains standard Maven configuration, including properties for compiler and target, build plugins for Surefire and Maven-Site, and dependencies on JUnit Jupiter.

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.example</groupId>
  <artifactId>homework</artifactId>
  <version>1.0-SNAPSHOT</version>

  <properties>
    <maven.compiler.source>8</maven.compiler.source>
    <maven.compiler.target>8</maven.compiler.target>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <surefire.version>3.5.5</surefire.version>
  </properties>

  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-surefire-plugin</artifactId>
        <version>${surefire.version}</version>
      </plugin>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-surefire-report-plugin</artifactId>
        <version>${surefire.version}</version>
      </plugin>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-site-plugin</artifactId>
        <version>3.12.1</version>
      </plugin>
    </plugins>
  </build>
  <reporting>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-surefire-report-plugin</artifactId>
        <version>${surefire.version}</version>
      </plugin>
    </plugins>
  </reporting>
  <dependencies>
    <dependency>
      <groupId>org.junit.jupiter</groupId>
      <artifactId>junit-jupiter-engine</artifactId>
      <version>6.8.2</version>
      <scope>test</scope>
    </dependency>
  </dependencies>

```

Test I – Stack LIFO

The screenshot shows the IntelliJ IDEA interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, Help.
- Toolbar:** Standard icons for file operations.
- Status Bar:** Fri Feb 27 3:37 PM, 8:25 LF, UTF-8, 4 spaces.
- Project View:** Shows the project structure under "homework".
- Code Editor:** The StackTest.java file is open, displaying JUnit test code for a Stack class.

```
1 import org.junit.jupiter.api.Test;
2
3 import java.util.Stack;
4
5 import static org.junit.jupiter.api.Assertions.assertEquals;
6 import static org.junit.jupiter.api.Assertions.assertTrue;
7
8 public class StackTest {
9     @Test
10    void stackShouldFollowLIFO() {
11        Stack<Integer> stack = new Stack<>();
12
13        // push() some elements
14        stack.push(14);
15        stack.push(56);
16        stack.push(0);
17        stack.push(89);
18        stack.push(14);
19
20        // ensure LIFO order on pop()
21        assertEquals(14, stack.pop());
22        assertEquals(89, stack.pop());
23        assertEquals(0, stack.pop());
24
25        // ensure if peek() works
26        assertEquals(56, stack.peek());
27
28        stack.pop();
29        stack.pop();
30
31        // ensure stack is empty
32        assertTrue(stack.isEmpty());
33    }
34}
```

Test II – subList() Shallow Copy

The screenshot shows the IntelliJ IDEA interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, Help.
- Toolbar:** Version Control, pom.xml (homework), StackTest.java, Person.java, SubListTest.java, OptionalTest.java, InstanceLifecycleTest.java.
- Project Structure:** homework (~Desktop/samidm), .idea, .mvn, src, main, java, Person, resources, test, target, classes, generated-sources, generated-test-sources, maven-status, reports, css, fonts, images, img, js, surefire.html, surefire-reports, InstanceLifecycleTest, OptionalTest.txt, StackTest.txt, SubListTest.txt, TEST-InstanceLifecycleTest, TEST-OptionalTest.txt, TEST-StackTest.xml, TEST-SubListTest.xml.
- Code Editor:** The code for SubListTest.java is displayed, demonstrating the shallow copy behavior of ArrayList's sublist method.
- Bottom Status Bar:** homework > src > test > java > SubListTest, 20:27 LF.

```
import org.junit.jupiter.api.Test;
import java.util.ArrayList;
import java.util.List;

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

/*
 * sublist() does not create a new independent list.
 * It returns a view backed by the original list.
 * - changes affect both (original & view).
 * - modifying an element has an effect on both.
 * - they share the same underlying data.
 *
 * Test case to prove sublist() is a shallow copy:
 * 1) Modifying the sublist modifies the original list.
 * 2) They should share the same object reference.
 */
public class SubListTest {
    @Test
    void sublistIsShallowCopy() {
        List<Person> original = new ArrayList<>();
        original.add(new Person(name: "Sam"));
        original.add(new Person(name: "Landon"));
        original.add(new Person(name: "Leo"));

        // create a sublist from original
        List<Person> subList = original.subList(0, 2);

        // modify the object inside sublist
        subList.get(1).name = "Updated Landon";

        // since it's a shallow copy, original should also reflect the change.
        assertEquals(expected: "Updated Landon", original.get(1).name);

        // verify same object reference
        assertSame(original.get(1), subList.get(1));
    }
}
```

The screenshot shows the IntelliJ IDEA interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, Help.
- Toolbar:** Version Control, pom.xml (homework), StackTest.java, Person.java, SubListTest.java, OptionalTest.java, InstanceLifecycleTest.java.
- Project Structure:** homework (~Desktop/samidm), .idea, .mvn, src, main, java, Person, resources, test, target, classes, generated-sources, generated-test-sources, maven-status, reports, css, fonts, images, img, js, surefire.html, surefire-reports, InstanceLifecycleTest, OptionalTest.txt, StackTest.txt, SubListTest.txt, TEST-InstanceLifecycleTest, TEST-OptionalTest.txt, TEST-StackTest.xml, TEST-SubListTest.xml.
- Code Editor:** The code for Person.java is displayed, showing the constructor definition.
- Bottom Status Bar:** homework > src > main > java > Person, 1:22 LF UTF-8 4 spaces.

```
public class Person {
    String name; 3 usages

    Person(String name) {
        this.name = name;
    }
}
```

Test III – Optional.of() NULL

The screenshot shows the IntelliJ IDEA interface with the following details:

- Project View:** On the left, the project structure for a Maven-based Java application named "homework" is displayed. It includes the main module, test modules, target directory, reports, and various configuration files like pom.xml and surefire-reports.
- Code Editor:** The central area shows the code for the `OptionalTest.java` file. The code uses JUnit Jupiter assertions to test the behavior of the `Optional.of(null)` constructor.
- Toolbars and Status Bar:** The top bar contains standard IntelliJ IDEA menu items and toolbars. The status bar at the bottom right shows the date and time as "Fri Feb 27 3:37 PM" and the line number "7:28 LF".

```
import org.junit.jupiter.api.Test;
import java.util.Optional;
import static org.junit.jupiter.api.Assertions.*;

public class OptionalTest {
    @Test
    void shouldThrowNPE() {
        NullPointerException exception = assertThrows(NullPointerException.class, () -> {
            Optional.of(null);
        });
        assertNotNull(exception);
    }

    @Test
    void shouldNotThrowNPE() {
        Optional<Object> optional = Optional.ofNullable(null);
        assertFalse(optional.isPresent());
    }
}
```

Test IV – Test Instance Lifecycle Method

IntelliJ IDEA

Fri Feb 27 3:37 PM

homework

Project

src

- main
- java
- Person
- resources

test

- target
- classes
- generated-sources
- generated-test-sources
- maven-status
- reports
- css
- fonts
- images
- img
- js
- surefire.html

surefire-reports

- InstanceLifecycleTest
- OptionalTest.txt
- StackTest.txt
- SubListTest.txt

TEST-InstanceLifecycleTest

TEST-OptionalTest.txt

TEST-StackTest.xml

TEST-SubListTest.xml

test-classes

.gitignore

pom.xml

External Libraries

Scratches and Consoles

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help

InstanceLifecycleTest.java

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

public class InstanceLifecycleTest {
    int counter = 0; 4 usages

    @Test
    void testOne() {
        counter++;
        System.out.println("testOne().this = " + this);
        assertEquals( expected: 1, counter);
    }

    @Test
    void testTwo() {
        counter++;
        System.out.println("testTwo().this = " + this);
        assertEquals( expected: 1, counter);
    }
}
```