

Task 1

GTv1 screenshot

The screenshot displays the IntelliJ IDEA IDE interface. The top toolbar shows the 'Run' button (a green play icon) and the 'Coverage' button (a green icon with a checkmark). The 'Project' view on the left shows the project structure, including the 'src' directory and the 'test' directory. The 'test' directory is expanded, showing the 'java' subdirectory, which contains the 'ee.ut.cs.swt.nextdate' package. The 'GTv1' test class is selected. The 'Coverage' view on the right shows the coverage for the 'GTv1' test class, with 100% class coverage and 87% line coverage. The 'Code' view in the center shows the source code of the 'GTv1' test class, which includes three test methods: 'testCreatesNextDate1', 'testCreatesNextDate0', and 'testCreatesNextDate3'. The 'Run' view at the bottom shows the test results for the 'GTv1' test suite, indicating that all 21 tests passed in 29 ms. The 'Run' view also shows the command line used to run the tests: 'C:\Users\kasnikov\.jdk\openjdk-21.0.2\bin\java.exe ...'. The 'Run' view also shows the coverage patterns used for the tests: 'include patterns: ee\ut\cs\swt\nextdate\..*', 'exclude patterns: .*Generated.*', and 'exclude annotations patterns: .*Generated.*'. The 'Run' view also shows the class transformation time: 'Class transformation time: 0.0164107s for 511 classes or 3.211487279843444E-5s per class'.

Project: NDv1 [nextdate] C:\Users\kasnikov\Desktop\All about coding\testing\hw5\NDv1

src

main

java

ee.ut.cs.swt.nextdate 100% classes, 87% lines covered

NextDate 100% methods, 87% lines covered

resources

test

java

ee.ut.cs.swt.nextdate

GTv1

MTv1

target

pom.xml

External Libraries

Scratches and Consoles

Package: ee.ut.cs.swt.nextdate

import ...

public class GTv1 {

@Test(timeout = 4000)

public void testCreatesNextDate1() throws Throwable {

NextDate nextDate0 = new NextDate(m: 10, d: 10, y: 10

String string0 = nextDate0.run(month: 10, day: 31, ye

assertEquals(expected: "11/1/1970", string0);

}

@Test(timeout = 4000)

public void testCreatesNextDate0() throws Throwable {

NextDate nextDate0 = new NextDate(m: 12, d: 12, y: 12

String string0 = nextDate0.run(month: 12, day: 1839,

assertEquals(expected: "invalid Input Date", string0);

}

@Test(timeout = 4000)

public void testCreatesNextDate3() throws Throwable {

NextDate nextDate0 = new NextDate(m: 9, d: 9, y: 9);

String string0 = nextDate0.run(month: 9, day: 31, year

assertEquals(expected: "Invalid Input Date", string0);

}

Coverage: GTv1

GTv1 (ee.ut.cs.swt.nextdate) 29 ms

testRunWithNegative 8 ms

testRunWithNegativeAndNegative 1 ms

testRunReturningNonEmptyString 0 ms

testRunWithPositive0 1 ms

testRunWithPositive1 1 ms

testRunWithPositive2 1 ms

testCreatesNextDate0 1 ms

testCreatesNextDate1 10 ms

testCreatesNextDate2 1 ms

testCreatesNextDate3 1 ms

Tests passed: 21 of 21 tests - 29 ms

C:\Users\kasnikov\.jdk\openjdk-21.0.2\bin\java.exe ...

IntelliJ IDEA coverage runner

Line coverage ...

include patterns:

ee\ut\cs\swt\nextdate\..*

exclude patterns:

exclude annotations patterns:

.*Generated.*

Class transformation time: 0.0164107s for 511 classes or 3.211487279843444E-5s per class

Process finished with exit code 0

NDv1 > src > test > java > ee > ut > cs > swt > nextdate > GTv1

6:15 CRLF UTF-8 Tab*

MTv1 screenshot

The screenshot displays the IntelliJ IDEA IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, and Help. The toolbar shows icons for running tests and viewing coverage.

The **Project** view on the left shows the project structure:

- NDv1 [nextdate] C:\Users\kasnikov\Desktop\All about coding\testing\hw5\NDv1
 - .idea
 - src
 - main
 - java
 - ee.ut.cs.swt.nextdate 100% classes, 65% lines covered
 - NextDate 100% methods, 65% lines covered
 - test
 - java
 - ee.ut.cs.swt.nextdate
 - GTv1
 - MTv1
 - target
 - pom.xml
 - External Libraries
 - Scratches and Consoles

The **Code** view in the center shows the `NextDate.java` file with the following code:

```
1 package ee.ut.cs.swt.nextdate;
2
3 import ...
4
5
6 public class GTv1 {
7
8     @Test(timeout = 4000)
9     public void testCreatesNextDate1() throws Throwable {
10         NextDate nextDate0 = new NextDate(m: 10, d: 10, y: 10);
11         String string0 = nextDate0.run(month: 10, day: 31, year: 1970);
12         assertEquals(expected: "11/1/1970", string0);
13     }
14
15     @Test(timeout = 4000)
16     public void testCreatesNextDate0() throws Throwable {
17         NextDate nextDate0 = new NextDate(m: 12, d: 12, y: 12);
18         String string0 = nextDate0.run(month: 12, day: 1839, year: 1970);
19         assertEquals(expected: "invalid Input Date", string0);
20     }
21
22     @Test(timeout = 4000)
23     public void testCreatesNextDate3() throws Throwable {
24         NextDate nextDate0 = new NextDate(m: 9, d: 9, y: 9);
25         String string0 = nextDate0.run(month: 9, day: 31, year: 1970);
26         assertEquals(expected: "invalid Input Date", string0);
27     }
28 }
```

The **Coverage** view on the right shows the following data:

Element	Class, %	Method, %	Line, %
ee.ut.cs.swt.nextdate	100% (1/1)	100% (6/6)	65% (27/41)
NextDate	100% (1/1)	100% (6/6)	65% (27/41)

The **Run** view at the bottom shows the test results for `MTv1`:

- MTv1 (ee.ut.cs.swt.nextdate) 21 ms
- testDecemberToJanuary 20 ms
- testLeapYear 1 ms
- testJuneToJuly 0 ms
- testMarchToApril 0 ms

The **Console** view shows the following output:

```
C:\Users\kasnikov\.jdk\openjdk-21.0.2\bin\java.exe ...
---- IntelliJ IDEA coverage runner ----
Line coverage ...
include patterns:
ee\ut\cs\swt\nextdate\.*
exclude patterns:
exclude annotations patterns:
.*Generated.*

org.junit.ComparisonFailure:
Expected: "invalid Input Date"
Actual: "11/1/1970"
```

Task 2, screenshots

GTv1

The screenshot displays the IntelliJ IDEA IDE interface with the following components:

- Project View:** Shows the project structure with the following hierarchy:
 - ee.ut.cs.swt.nextdate (100% classes, 81% lines covered)
 - NextDate (100% methods, 81% lines covered)
 - resources
 - test
 - java
 - ee.ut.cs.swt.nextdate
 - GTv1 (selected)
 - GTv12
 - MTv1
 - target
- Structure View:** Lists the methods in the selected `GTv1` class:
 - `testCreatesNextDate1(): void`
 - `testCreatesNextDate0(): void`
 - `testCreatesNextDate3(): void`
 - `testCreatesNextDate2(): void`
 - `testCreatesNextDate5(): void`
 - `testCreatesNextDate4(): void`
 - `testRunReturningNonEmptyString(): void`
 - `testRunWithNegative(): void`
 - `testRunWithNegativeAndNegative(): void`
- Code Editor:** Displays the `NextDate.java` file with the following code:

```
182 String string0 = nextDate0.run( month: 1, day: 1, year: 1 );
183 assertEquals( expected: "invalid Input Date", string0 );
184 }
185
186 @Test(timeout = 4000)
187 public void testCreatesNextDate7() throws Throwable {
188     NextDate nextDate0 = new NextDate( m: 7, d: 7, y: 7 );
189     String string0 = nextDate0.run( month: 7, day: 7, year: 7 );
190     assertEquals( expected: "7/7/1938", string0 );
191 }
192
193 @Test(timeout = 4000)
194 public void testCreatesNextDate6() throws Throwable {
195     NextDate nextDate0 = new NextDate( m: 12, d: 12, y: 12 );
196     String string0 = nextDate0.run( month: 12, day: 12, year: 12 );
197     assertEquals( expected: "12/13/1839", string0 );
198 }
199
200 @Test(timeout = 4000)
201 public void testCreatesNextDate9() throws Throwable {
202     NextDate nextDate0 = new NextDate( m: 2, d: 2, y: 2 );
203     String string0 = nextDate0.run( month: 2, day: 28, year: 2 );
204     assertEquals( expected: "3/1/1879", string0 );
205 }
```
- Coverage View:** Shows the coverage for the selected `GTv1` class:

Element	Class, %	Method, %	Line, %
ee.ut.cs.swt.nextdate	100% (1/1)	100% (7/7)	81% (40/49)
NextDate	100% (1/1)	100% (7/7)	81% (40/49)
- Test Results:** Shows the results of the tests:
 - GTv1 (ee.ut.cs.swt.nextdate) 23 ms
 - testRunWithNegative 8 ms
 - testRunWithNegativeAndNegative 1 ms
 - testRunReturningNonEmptyString 1 ms
 - testRunWithPositive0 1 ms
 - testRunWithPositive1 0 ms
 - testRunWithPositive2 0 ms
 - testCreatesNextDate0 0 ms
 - testCreatesNextDate1 8 ms
 - testCreatesNextDate2 0 ms
 - testCreatesNextDate3 0 ms
- Output Console:** Displays the output of the tests:

```
C:\Users\kasnikov\.jdk\openjdk-21.0.2\bin\java.exe ...
---- IntelliJ IDEA coverage runner ----
Line coverage ...
include patterns:
ee\ut\cs\swt\nextdate\.*
exclude patterns:
.*Generated.*
Class transformation time: 0.0175093s for 511 classes or 3.4264774951076315E-5s per class
OnProcess finished with exit code 0
```

GTv12

The screenshot displays the IntelliJ IDEA IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, and Help. The toolbar shows icons for running and debugging. The main editor area is split into three panes:

- Project View:** Shows the project structure with folders for `ee.ut.cs.swt.nextdate`, `resources`, `test`, and `target`. The `test` folder contains `java` and `ee.ut.cs.swt.nextdate` subfolders. The `ee.ut.cs.swt.nextdate` folder contains `GTv1`, `GTv12`, and `MTv1` classes.
- Structure View:** Shows the class structure for `GTv12`, listing methods like `testCreatesNextDate1()`, `testCreatesNextDate0()`, `testCreatesNextDate3()`, `testCreatesNextDate2()`, `testCreatesNextDate5()`, `testCreatesNextDate4()`, `testRunWithNegative()`, `testCreatesNextDate14()`, and `testCreatesNextDate13()`.
- Code Editor:** Displays the source code for `GTv12.java`. The code includes package declarations, imports, and several test methods using `@Test` and `assertEquals`.

The **Coverage** pane on the right shows the coverage for `GTv12` and `NextDate`. The coverage table is as follows:

Element	Class, %	Method, %	Line, %
ee.ut.cs.swt.nextdate	100% (1/1)	100% (7/7)	97% (48/49)
NextDate	100% (1/1)	100% (7/7)	97% (48/49)

The **Cover** pane at the bottom shows the test results for `GTv12`. The tests passed, and the coverage runner output is displayed:

```
Tests passed: 26 of 26 tests - 35 ms
C:\Users\kasnikov\.jdk\openjdk-21.0.2\bin\java.exe ...
---- IntelliJ IDEA coverage runner ----
Line coverage ...
include patterns:
ee\ut\cs\swt\nextdate\.*
exclude patterns:
exclude annotations patterns:
.*Generated.*
Class transformation time: 0.0168506s for 511 classes or 3.297573385518591E-5s per class
Process finished with exit code 0
```

MTv1

FileEditViewNavigateCodeRefactorBuildRunToolsVCSWindowHelp

NDv1Version control

MTv1

Project

ee.ut.cs.swt.nextdate 100% classes, 65% lines covered

NextDate 100% methods, 65% lines covered

resources

test

java

ee.ut.cs.swt.nextdate

GTv1

GTv12

MTv1

target

Structure

MTv1

tearDownAfterClass(): void

testJuneToJuly(): void

testMarchToApril(): void

testDecemberToJanuary(): void

testLeapYear(): void

trialDate: NextDate = new NextDate(...)

NextDate.java

GTv12.java

GTv1.java

MTv1.java

Coverage

MTv1

13

14

15

16

17

18

19

20

21

24

25

26

29

30

31

32

33

34

37

38

//@Test

/*

* Test method for 'ee.ut.cs.swt.nextdate.NextDate.run(int

*/

@Test

public final void testJuneToJuly() { assertEquals(expected:

@Test

public final void testMarchToApril() { assertEquals(expecte

@Test

public final void testDecemberToJanuary() { assertEquals(e

@Test

public final void testLeapYear() { assertEquals(expected: "

}

Element

Class, %

Method, %

Line, % ^

ee.ut.cs.swt.nextdate

NextDate

100% (1/1)

100% (7/7)

65% (32/49)

100% (1/1)

100% (7/7)

65% (32/49)

Cover

MTv1

MTv1 (ee.ut.cs.swt.nextdate) 12ms

testDecemberToJanuary 2ms

testLeapYear 10ms

testJuneToJuly 0ms

testMarchToApril 0ms

Tests passed: 4 of 4 tests - 12ms

C:\Users\kasnikov\.jdk\openjdk-21.0.2\bin\java.exe ...

---- IntelliJ IDEA coverage runner ----

Line coverage ...

include patterns:

ee\ut\cs\swt\nextdate\..*

exclude patterns:

exclude annotations patterns:

.*Generated.*

Class transformation time: 0.0132703s for 499 classes or 2.6593787575150303E-5s per class

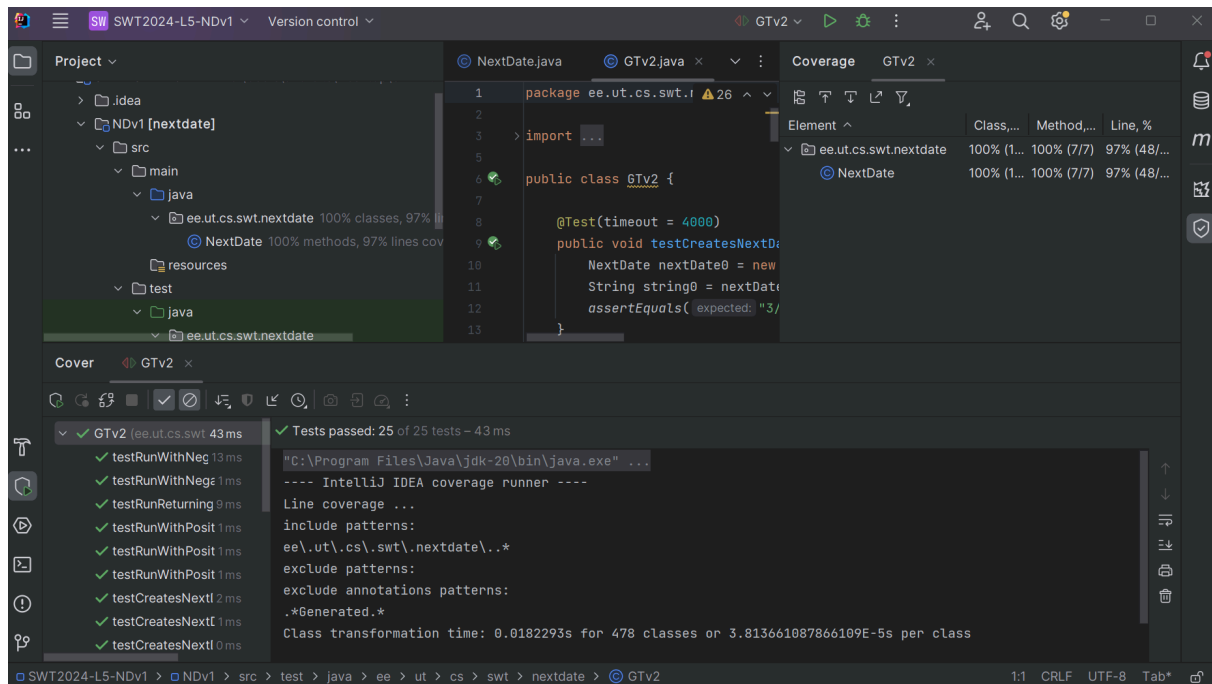
Done:ace finished with exit code 0

NDv1 > src > test > java > ee > ut > cs > swt > nextdate > MTv1

38:1 CRLF UTF-8 Tab*

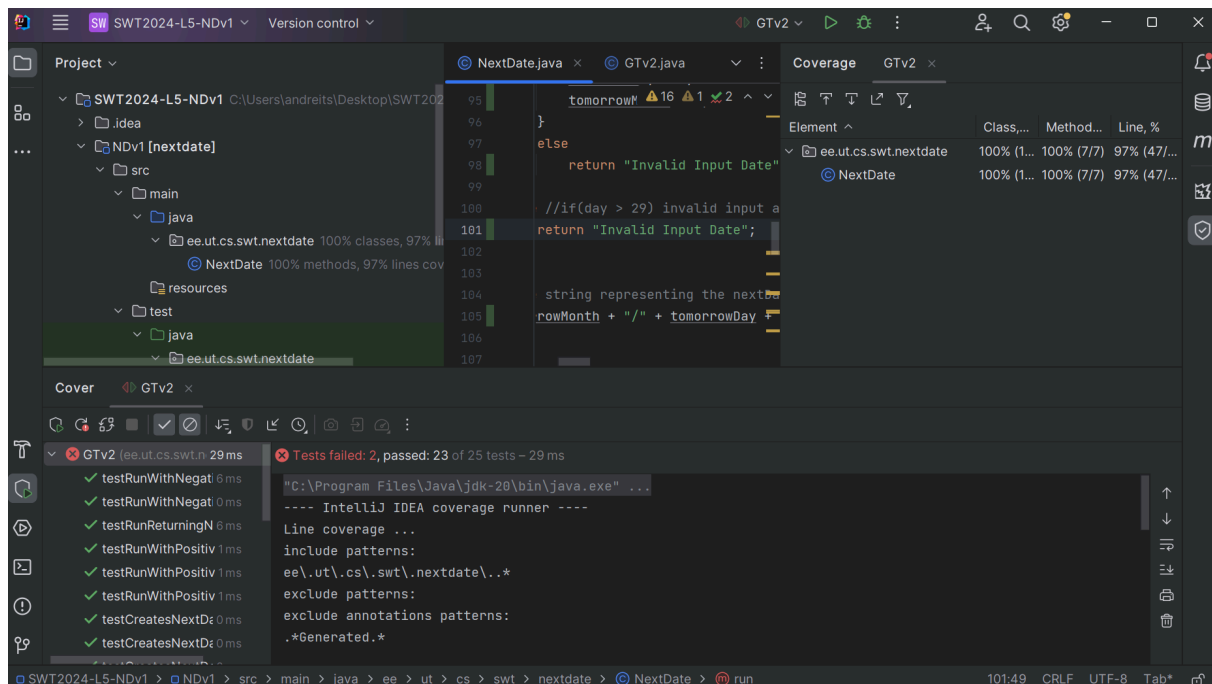
Task 3

GTv2



Task 4

GTv2



MTv1

The screenshot displays the IntelliJ IDEA IDE interface. The top toolbar shows the 'Run' button (a green play icon) and the 'Coverage' button (a magnifying glass icon). The 'Project' view on the left shows the project structure with the following hierarchy:

- resources
 - test
 - java
 - ee.ut.cs.swt.nextdate
 - GTv1
 - GTv2
 - GTv12
 - MTv1
- target
- pom.xml
- External Libraries
- Scratches and Consoles

The 'Coverage' view on the right shows the following data:

Element	Class, %	Method, %	Line, %
ee.ut.cs.swt.nextdate	100% (1/1)	100% (7/7)	70% (34/48)
NextDate	100% (1/1)	100% (7/7)	70% (34/48)

The 'Cover' view at the bottom shows the test results for MTv1:

- MTv1 (ee.ut.cs.swt.nextdate) 14 ms
 - testDecemberToJanuary 2 ms
 - testLeapYear 5 ms
 - testJuneToJuly 7 ms
 - testMarchToApril 0 ms

The 'Tests failed: 1, passed: 3 of 4 tests - 14 ms' message is displayed. The console output shows the following text:

```
"C:\Program Files\Java\jdk-20\bin\java.exe" ...  
---- IntelliJ IDEA coverage runner ----  
Line coverage ...  
include patterns:  
ee\ut\cs\swt\nextdate\.*  
exclude patterns:  
exclude annotations patterns:  
.*Generated.*
```

Task 5

InMemorySalesSystemDaoT

The screenshot displays the IntelliJ IDEA IDE with the following components:

- Project View:** Shows the project structure with folders like `main`, `test`, and `resources`. The `test` folder contains `InMemorySalesSystemDAOT`.
- Code Editor:** Displays the `InMemorySalesSystemDAO.java` file. It includes a comment about transaction-related methods and a `public interface SalesSystemDAO` with several methods: `findStockItems()`, `findStockItem(Long id)`, `saveStockItem(StockItem stockItem)`, `saveSoldItem(SoldItem item)`, `beginTransaction()`, and `rollbackTransaction()`.
- Coverage View:** Shows the coverage for `InMemorySalesSystemDAO`. The table below summarizes the coverage data:
- Test Runner:** Shows the test results for `InMemorySalesSystemDAOT`. All 7 tests passed.

Element	Class, %	Method, %	Line, %
ee.ut.math.tvt.salessystem.dao	100% (1/1)	100% (8/8)	100% (19/19)
InMemorySalesSystemDAO	100% (1/1)	100% (8/8)	100% (19/19)
SalesSystemDAO	100% (0/0)	100% (0/0)	100% (0/0)

Test Results:

- testSaveSoldItem: 14 ms
- testFindStockItems: 1 ms
- testFindStockItemReturningNull: 1 ms
- testRollbackTransaction: 1 ms
- testCommitTransaction: 1 ms
- testSaveStockItem: 1 ms
- testBeginTransaction: 0 ms

Tests passed: 7 of 7 tests - 19 ms

IntelliJ IDEA coverage runner output:

```
C:\Users\kasnikov\.jdk\openjdk-21.0.2\bin\java.exe ...
---- IntelliJ IDEA coverage runner ----
Line coverage ...
include patterns:
ee\ut\math\tvt\salessystem\dao\.*
exclude patterns:
exclude annotations patterns:
.*generated.*
Class transformation time: 0.0141761s for 533 classes or 2.659681050656604E-5s per class
Process finished with exit code 0
```


ShoppingCartT

The screenshot displays the IntelliJ IDEA IDE with the following components:

- Project View:** Shows the project structure with folders for `main`, `test`, and `resources`. The `test` folder contains `java` and `target` subfolders. The `java` folder contains `InMemorySalesSystemDAO`, `ShoppingCartT`, `SoldItemT`, and `StockItemT`.
- Code Editor:** Displays the `SalesSystemDAO.java` file. The code includes a `public interface SalesSystemDAO` with methods `findStockItems()`, `findStockItem(long id)`, `saveStockItem(StockItem stockItem)`, `saveSoldItem(SoldItem item)`, `beginTransaction()`, and `rollbackTransaction()`.
- Coverage Table:** Shows the coverage for the `ShoppingCartT` class. The table has columns for `Element`, `Class, %`, `Method, %`, and `Line, %`.
- Run/Debug Console:** Shows the output of the test run, including the command `C:\Users\kasnikov\.jdk\openjdk-21.0.2\bin\java.exe ...` and the coverage report.

Element	Class, %	Method, %	Line, %
ee.ut.math.tvt.salessystem.logic	100% (1/1)	100% (5/5)	81% (13/16)
ShoppingCart	100% (1/1)	100% (5/5)	81% (13/16)

Tests passed: 4 of 4 tests - 15 ms

Line coverage ...

include patterns:

ee\ut\math\..tvt\..salessystem\..logic\..*

exclude patterns:

exclude annotations patterns:

.*Generated.*

Class transformation time: 0.0150242s for 531 classes or 2.829416195856874E-5s per class

Process finished with exit code 0

SoldItemT

The screenshot shows an IDE with the following components:

- Project Explorer:** Displays a project structure with packages like `main`, `ee.ut.math.tvt.salessystem`, `dao`, `dataobjects`, `logic`, `resources`, and `test`. The `test` package contains `InMemorySalesSystemDAOT`, `ShoppingCartT`, `SoldItemT`, and `StockItemT`.
- Code Editor:** Shows the `SalesSystemDAO.java` file. It contains a comment block and a public interface `SalesSystemDAO` with methods: `findStockItems()`, `findStockItem(long id)`, `saveStockItem(StockItem stockItem)`, `saveSoldItem(SoldItem item)`, and `beginTransaction()`.
- Coverage:** A table showing coverage for `ee.ut.math.tvt.salessystem.dataobjects`.

Element	Class, %	Method, %	Line, %
<code>ee.ut.math.tvt.salessystem.dataobjects</code>	100% (2/2)	62% (17/27)	58% (21/36)
<code>SoldItem</code>	100% (1/1)	100% (14/14)	100% (18/18)
<code>StockItem</code>	100% (1/1)	23% (3/13)	16% (3/18)
- Test Runner:** Shows a list of tests for `SoldItemT`. The test `testGetSum` is highlighted with a red icon, indicating it failed.

Test Name	Duration
<code>testGetStockItem</code>	14 ms
<code>testGetPrice</code>	0 ms
<code>testGetId</code>	3 ms
<code>testSetId</code>	1 ms
<code>testToString</code>	1 ms
<code>testGetName</code>	1 ms
<code>testSetStockItem</code>	0 ms
<code>testSetQuantity</code>	0 ms
<code>testGetQuantity</code>	0 ms
<code>testGetSum</code>	13 ms
<code>testSetName</code>	0 ms
<code>testSetPrice</code>	0 ms
- Test Output:** Displays the error message for the failed test: `java.lang.NullPointerException: Cannot invoke "java.lang.Integer.intValue()" because "this.quantity" is null`. The stack trace points to `at ee.ut.math.tvt.salessystem.dataobjects.SoldItem.getSum(SoldItem.java:59)` and `at ee.ut.math.tvt.salessystem.dataobjects.SoldItemT.testGetSum(SoldItemT.java:72) <10 internal lines>`.
- Status Bar:** Shows the file path `POS-v1.1 > src > test > java > SoldItemT` and the current encoding `UTF-8` with `4 spaces`.

StockItemT

The screenshot displays the IntelliJ IDEA IDE interface. The main editor shows the `StockItemT.java` file with the following code:

```
1 package ee.ut.math.tvt.salessystem.dataobjects;
2
3 import ...
4
5 public class StockItemT {
6
7
8     @Test(timeout = 4000)
9     public void testGetId() throws Throwable {
10         Long long0 = Long.valueOf(899L);
11         StockItem stockItem0 = new StockItem(long0, name: "",
12         stockItem0.getId());
13         assertEquals( expected: 899.0, stockItem0.getPrice(), d
14         assertEquals( expected: "$", stockItem0.getDescription()
15         assertEquals( expected: 204, stockItem0.getQuantity());
16         assertEquals( expected: "", stockItem0.getName());
17     }
18
19     @Test(timeout = 4000)
20     public void testSetId() throws Throwable {
21         Long long0 = Long.valueOf(899L);
22         StockItem stockItem0 = new StockItem(long0, name: "",
23         stockItem0.setId(long0);
```

The Project tool window on the left shows the project structure:

- main
 - java
 - ee.ut.math.tvt.salessystem
 - dao
 - InMemorySalesSystemDAO
 - SalesSystemDAO
 - dataobjects (50% classes, 50% lines covered)
 - SoldItem (0% methods, 0% lines covered)
 - StockItem (100% methods, 100% lines covered)
 - logic
 - ShoppingCart
 - resources
 - test
 - java
 - InMemorySalesSystemDAO
 - ShoppingCart
 - SoldItem
 - StockItem
- target
- pom.xml

The Coverage tool window on the right shows the coverage report for `StockItemT`:

Element	Class, %	Method, %	Line, %
ee.ut.math.tvt.salessystem.dataobjects	50% (1/2)	48% (13/27)	50% (18/36)
SoldItem	0% (0/1)	0% (0/14)	0% (0/18)
StockItem	100% (1/1)	100% (13/13)	100% (18/18)

The Run tool window at the bottom shows the test results for `StockItemT`:

✓ Tests passed: 11 of 11 tests – 23 ms

- ✓ testGetPrice 15 ms
- ✓ testGetId 1 ms
- ✓ testSetId 1 ms
- ✓ testSetDescription 0 ms
- ✓ testToString 1 ms
- ✓ testGetName 1 ms
- ✓ testSetQuantity 1 ms
- ✓ testGetQuantity 0 ms
- ✓ testGetDescription 1 ms
- ✓ testSetName 1 ms
- ✓ testSetPrice 1 ms

The output of the test run is as follows:

```
C:\Users\kasnikov\.jdk\openjdk-21.0.2\bin\java.exe ...
---- IntelliJ IDEA coverage runner ----
Line coverage ...
include patterns:
ee\ut\math\tvt\salessystem\dataobjects\.*
exclude patterns:
exclude annotations patterns:
.*Generated.*
Class transformation time: 0.0137722s for 527 classes or 2.6133206831119545E-5s per class
Process finished with exit code 0
```