

# Task 1

```

public class GTv1 {
    @Test(timeout = 4000) new *
    public void testCreatesNextDate1() throws Throwable {
        NextDate nextDate0 = new NextDate((-382), (-382), (-382));
        String string0 = nextDate0.run( month: 1, day: 2, year: 1847);
        assertEquals( expected: "1/3/1847", string0);
    }

    @Test(timeout = 4000) new *
    public void testCreatesNextDate0() throws Throwable {
        NextDate nextDate0 = new NextDate( m: 8, d: 1813, y: 8);
        String string0 = nextDate0.run( month: 8, day: 1813, year: 1813);
    }
}

```

Coverage GTv1 x

Elem...	Class, %	Method, %	Line, %	Branch, %
ee.ut.cs.swt.r	100% (1/1)	100% (7/7)	97% (48/49)	96% (62/64)
1	100% (1/1)	100% (7/7)	97% (48/49)	96% (62/64)

Tests passed: 27 of 27 tests - 45 ms

```

NextDate nextDate0 = new NextDate( m: 12, d: 31, y: 20
String string0 = nextDate0.run( month: 12, day: 31, year: 20
assertEquals( expected: "Invalid Next Year", string0);
}

@Test(timeout = 4000) new *
public void testCreatesNextDate4() throws Throwable {
    NextDate nextDate0 = new NextDate( m: 2, d: 28, y: 200
    String string0 = nextDate0.run( month: 2, day: 28, year: 200
    assertEquals( expected: "2/29/2004", string0);
}

```

Coverage MTv1 x

Elem...	Class, %	Method, %	Line, %	Branch, %
ee	100% (1/1)	100% (7/7)	59% (29/49)	53% (34/64)
1	100% (1/1)	100% (7/7)	59% (29/49)	53% (34/64)

Tests failed: 1, passed: 3 of 4 tests - 45 ms

org.junit.ComparisonFailure:  
Expected :Invalid Next Year

Table 1

Variable / Output	EC	Description	Covered by MTv1	Covered by GTv1
Month	M1	Valid month (1–12)	Yes	Yes
	M2	Invalid month (<1 or >12)	No	Yes
Day	D1	Valid day for given month and year	Yes	Yes
	D2	Invalid day (e.g., 2/30, negative)	No	Yes

<b>Year</b>	Y1	Valid year (e.g., 1801–2021)	Yes	Yes
	Y2	Invalid year (<1801 or >2021)	No	Yes
<b>Output</b>	O1	Valid next day in same month	Yes	Yes
	O2	Month change (e.g., 1/31 → 2/1)	Yes	Yes
	O3	Year change (e.g., 12/31 → 1/1)	Yes	Yes
	O4	"Invalid Input Date"	No	Yes
	O5	"Invalid Next Year"	No (test failed)	Yes
	O6	Invalid output (e.g., 12/32/1915)	No	Yes

**Table 2**

<b>Aspect</b>	<b>MTv1</b>	<b>GTv1</b>
<b>Number of tests</b>	4	27
<b>Tests passed</b>	3	27
<b>Tests failed</b>	1 (leap year test)	0
<b>Input EC coverage</b>	Only valid inputs	Valid and invalid inputs
<b>Output EC coverage</b>	O1, O2, O3	O1, O2, O3, O4, O5, O6
<b>Detected failures</b>	No actual detection (test failed due to wrong expectation)	Detected edge case outputs

<b>Code coverage – Lines</b>	59%	97%
<b>Code coverage – Branches</b>	53%	96%
<b>Code coverage – Methods</b>	100%	100%
<b>EC coverage completeness</b>	Partial	Comprehensive
<b>Defect detection effectiveness</b>	Low (missed logic)	High (validated corner cases)

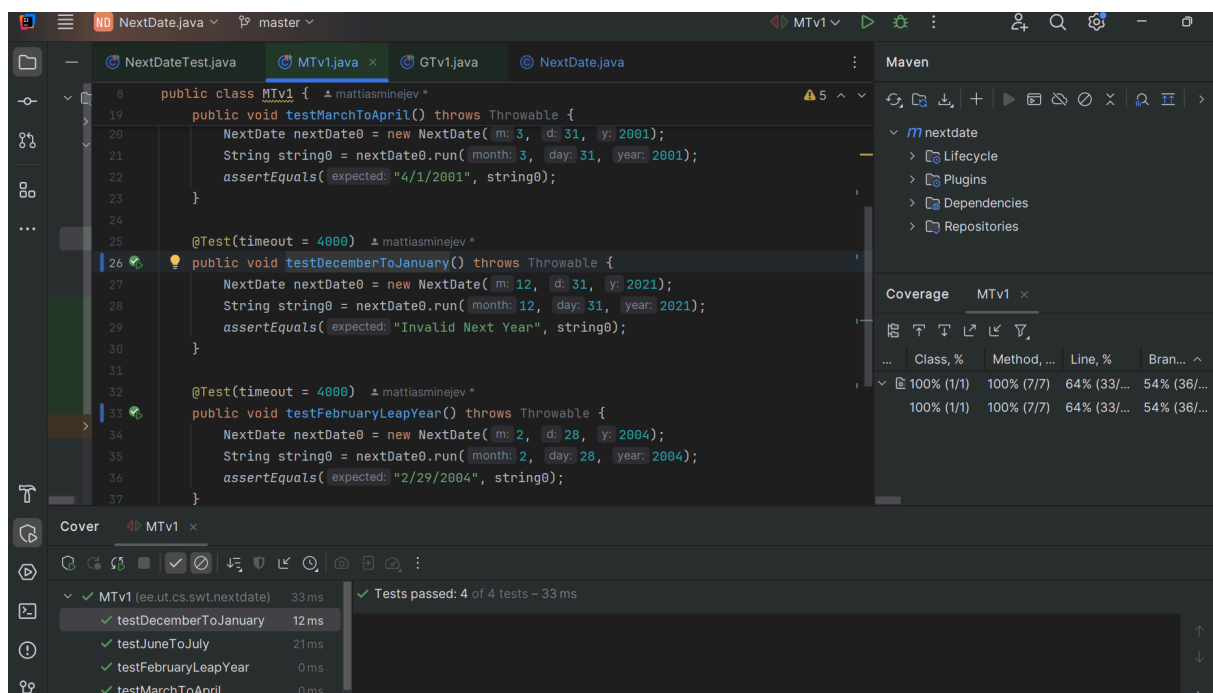
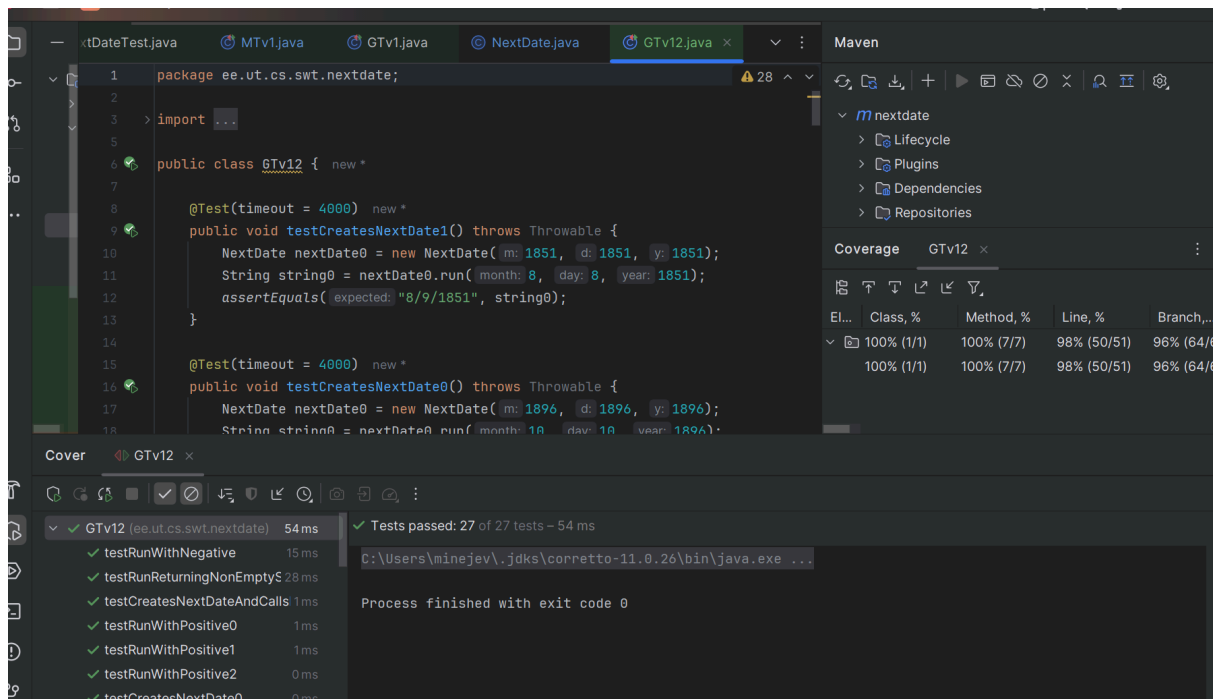
## Task 2

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

- Code Editor:** Shows the `GTV1` class with methods `testCreatesNextDate8`, `testCreatesNextDate10`, and `testCreatesNextDate12`. The code includes assertions and date calculations.
- Maven Tab:** Shows the project structure with coverage for `nextdate`, `Lifecycle`, `Plugins`, and `Dependencies`.
- Coverage Tab:** Displays a table with coverage metrics for the `GTV1` class.
 

El...	Class	%	Method	%	Line	%	Branch	%
100%	100%	100%	100%	100%	96%	95%	96%	95%
- Cover Tab:** Shows a list of tests and their results.
 

Test Name	Result	Duration
testRunWithNegative	Failed	76 ms
testRunWithNegativeAndNegr	Failed	13 ms
testRunReturningNonEmptyS	Failed	48 ms
testCreatesNextDateAndCalls	Failed	4 ms
testRunWithPositive0	Passed	0 ms
testRunWithPositive1	Passed	0 ms
testRunWithPositive2	Passed	0 ms
testCreatesNextDate0	Passed	0 ms
testCreatesNextDate1	Passed	0 ms
testCreatesNextDate2	Passed	1 ms
- Error Message:** `org.junit.ComparisonFailure: Expected :Invalid Next Year Actual :Invalid Input Date`



GTV1 made tests with wrong NextDate code therefore also the test inputs and expected outputs were wrong. Obviously GTv12 is better because it makes tests with corrected code.

## Task3

The screenshot shows the IntelliJ IDEA IDE with the `GTV2.java` file open. The file is located in the `ee.ut.cs` package. The code defines a `GTV2` class with two test methods: `testCreates` and `testCreatesNDv2`. The `testCreates` method creates a `NDv2` object and asserts its `string0` property. The `testCreatesNDv2` method creates a `NDv2` object and asserts its `string0` property.

The Coverage window shows the following data:

Element	Class, %	Method, %	Line, %	Branch, %
ee	50% (1/2)	50% (7/14)	49% (50/102)	48% (64/132)
GTV2	0% (0/1)	0% (0/7)	0% (0/51)	0% (0/66)
GTV2 (1/1)	100% (1/1)	100% (7/7)	98% (50/51)	96% (64/66)

The Cover window shows the test results for `GTV2` (ee.ut.cs.swt.nextdate):

- testCreatesNDv2AndCallsRu: 38 ms
- testRunWithNegativeAndRunV: 1 ms
- testCreatesNDv20: 1 ms
- testCreatesNDv21: 1 ms
- testCreatesNDv22: 1 ms
- testCreatesNDv23: 0 ms
- testCreatesNDv24: 0 ms
- testCreatesNDv25: 0 ms

Tests passed: 26 of 26 tests - 49 ms

The screenshot shows the IntelliJ IDEA IDE with the `MTv1.java` file open. The file is located in the `ee.ut.cs` package. The code defines a `MTv1` class with two test methods: `testJuneToJuly` and `testDecemberToJanuary`. The `testJuneToJuly` method creates a `NextDate` object and asserts its `string0` property. The `testDecemberToJanuary` method creates a `NextDate` object and asserts its `string0` property.

The Coverage window shows the following data:

Element	Class, %	Method, %	Line, %	Branch, %
ee	100% (1/1)	100% (7/7)	64% (33/51)	54% (36/66)
MTv1	100% (1/1)	100% (7/7)	64% (33/51)	54% (36/66)

The Cover window shows the test results for `MTv1` (ee.ut.cs.swt.nextdate):

- testDecemberToJanuary: 15 ms
- testJuneToJuly: 25 ms
- testFebruaryLeapYear: 0 ms
- testMarchToApril: 1 ms

Tests passed: 4 of 4 tests - 41 ms

Due to `isLeapYear` already uncommented and working, both test files worked to perfection and all tests passed.