

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

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)

Cover GTv1

Tests passed: 27 of 27 tests - 45 ms

Process finished with exit code 0

```

NextDate nextDate0 = new NextDate( m: 12, d: 31, y: 20
String string0 = nextDate0.run( month: 12, day: 31, year: 2004);
assertEquals( expected: "Invalid Next Year", string0);
}

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

```

Coverage MTv1

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)

Cover MTv1

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

Year	Y1	Valid year (e.g., 1801–2021)	Yes	Yes
	Y2	Invalid year (<1801 or >2021)	No	Yes
Output	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

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

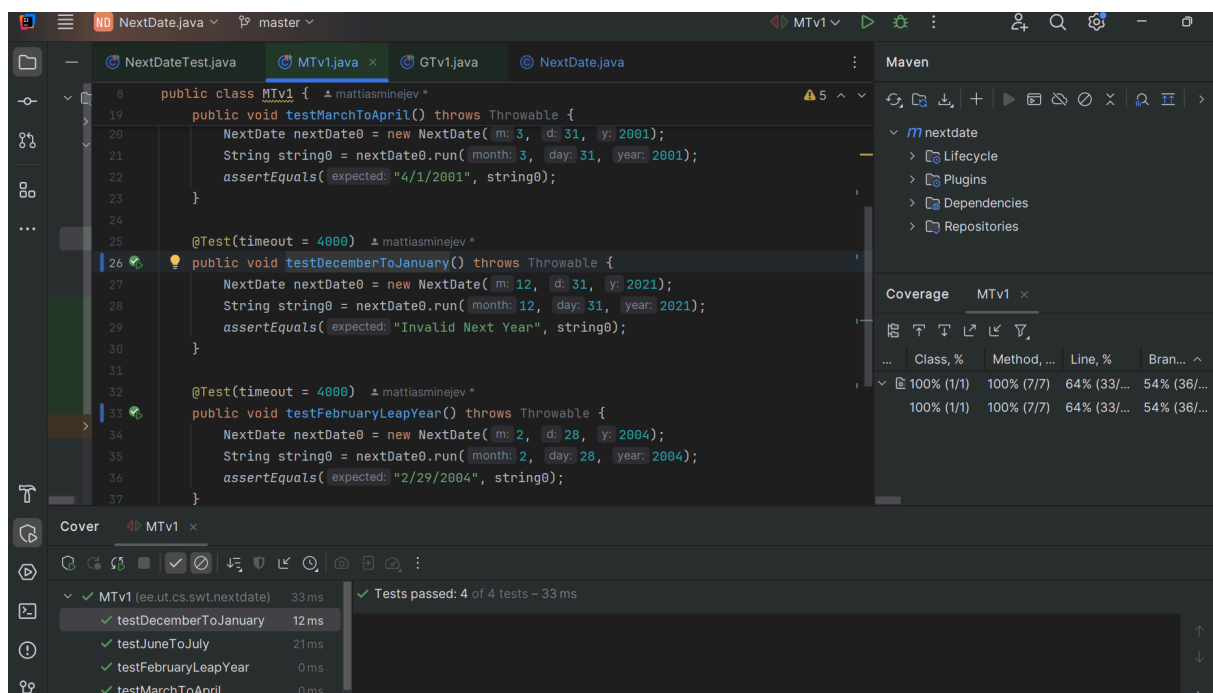
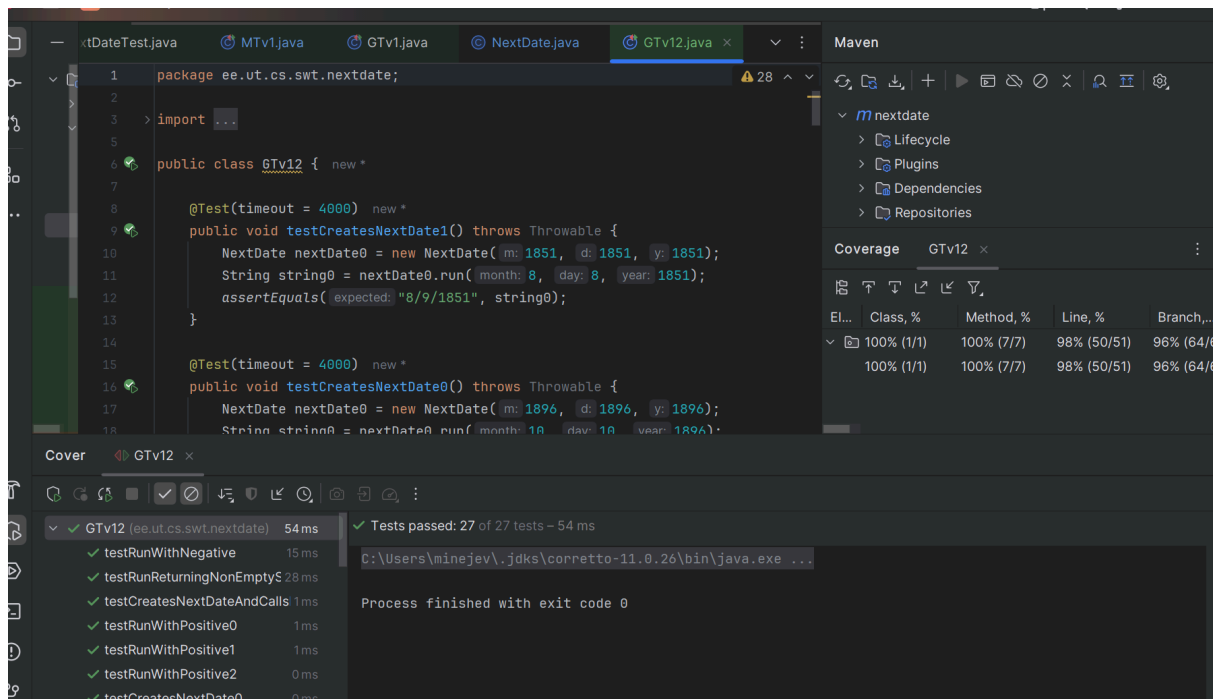
Code coverage – Lines	59%	97%
Code coverage – Branches	53%	96%
Code coverage – Methods	100%	100%
EC coverage completeness	Partial	Comprehensive
Defect detection effectiveness	Low (missed logic)	High (validated corner cases)

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 `GTV1`.

El...	Class	%	Method	%	Line	%	Branch	%
100%	100%	100%	100%	100%	96%	95%	96%	95%
- Run Tab:** Shows test results for `GTV1` (ee.ut.cs.swt.nextdate). The tests are:
 - `testRunWithNegative` (13 ms)
 - `testRunWithNegativeAndNegr` (0 ms)
 - `testRunReturningNonEmptyS` (48 ms)
 - `testCreatesNextDateAndCalls` (4 ms)
 - `testRunWithPositive0` (0 ms)
 - `testRunWithPositive1` (0 ms)
 - `testRunWithPositive2` (0 ms)
 - `testCreatesNextDate0` (0 ms)
 - `testCreatesNextDate1` (0 ms)
 - `testCreatesNextDate2` (1 ms)
- Error Details:** The failed tests show a `org.junit.ComparisonFailure` with the message "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 package `ee.ut.cs` and contains two test methods: `testCreates` and `testCreatesNDv2`. The coverage report for `GTV2` is displayed on the right, showing 100% coverage for all elements. The bottom panel shows the test results for `GTV2`, indicating that all 26 tests passed.

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 (100%)	100% (1/1)	100% (7/7)	98% (50/51)	96% (64/66)

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 package `ee.ut.cs.swt.nextdate` and contains four test methods: `testDecemberToJanuary`, `testJuneToJuly`, `testFebruaryLeapYear`, and `testMarchToApril`. The coverage report for `MTv1` is displayed on the right, showing 100% coverage for all elements. The bottom panel shows the test results for `MTv1`, indicating that all 4 tests passed.

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)

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.

Task4.1

The screenshot displays an IDE with two windows. The top window shows the source code for `GTV2.java` with two test methods: `testCreatesNDv220` and `testCreatesNDv25`. The bottom window shows the test results for `MTV2`, indicating that 1 test failed and 25 tests passed. The failed test is `testCreatesNDv220`, which failed due to a `org.junit.ComparisonFailure` where the expected value was `Invalid Input Date` and the actual value was `2/29/1900`. The error message includes a link to see the difference and the location of the failure in the code.

Test Results for MTV2:

Test Name	Duration	Status
testCreatesNDv214	0 ms	Passed
testCreatesNDv215	1 ms	Passed
testCreatesNDv216	1 ms	Passed
testCreatesNDv217	1 ms	Passed
testCreatesNDv218	1 ms	Passed
testCreatesNDv219	0 ms	Passed
testCreatesNDv220	9 ms	Failed
testRunWithNegativeAndRunV	0 ms	Passed
testRunWithPositive	0 ms	Passed

Error Message:

```
org.junit.ComparisonFailure:
Expected :Invalid Input Date
Actual   :2/29/1900
<Click to see difference>
```

Code Snippets:

GTV2.java:

```
public class GTv2 {
    @Test(timeout = 4000) new *
    public void testCreatesNDv220() throws Throwable {
        NDv2 nDv2_0 = new NDv2( m: 29, d: 29, y: 29);
        String string0 = nDv2_0.run( month: 2, day: 29, year: 1900);
        assertEquals( expected: "Invalid Input Date", string0);
    }

    @Test(timeout = 4000) new *
    public void testCreatesNDv25() throws Throwable {
        NDv2 nDv2_0 = new NDv2( m: 2, d: 2, y: 2);
        String string0 = nDv2_0.run( month: 2, day: 1945, year: 1945);
        assertEquals( expected: "Invalid Input Date", string0);
    }
}
```

NDv2.java:

```
public class NDv2 {
    public String run(int month, int day, int year) {
        if(isLeapYear(year)){ //AND a leap year - reset the day to 1, mont
            tomorrowDay = 1;
            tomorrowMonth = 3;
        }
        else
            return "Invalid Input Date";
    }
    else if(day > 29) //invalid input as February will never have more than
        return "Invalid Input Date";
    }
    //return the string representing the nextDate, in the form MM/DD/YY
    return tomorrowMonth + "/" + tomorrowDay + "/" + tomorrowYear;
}
```

One test failed. it failed due to the method not being able to bring out error: "Invalid Input Date".

Task 4.2

The screenshot shows an IDE with the file `MTv1.java` open. The code defines a class `MTv1` with two test methods: `testJuneToJuly()` and `testMarchToApril()`. The `testJuneToJuly()` method creates a `NextDate` object for June 30, 2001, and expects the next date to be July 1, 2001. The `testMarchToApril()` method is also present but its implementation is partially obscured. The coverage report on the right shows 50% class coverage, 50% method coverage, 32% line coverage, and 27% branch coverage. The test runner at the bottom indicates that 4 out of 4 tests passed in 46 ms.

```
package ee.ut.cs.swt.nextdate;

import ...

public class MTv1 {

    @Test(timeout = 4000)
    public void testJuneToJuly() throws Throwable {
        NextDate nextDate0 = new NextDate(m: 6, d: 30, y: 2001);
        String string0 = nextDate0.run(month: 6, day: 30, year: 2001);
        assertEquals(expected: "7/1/2001", string0);
    }

    @Test(timeout = 4000)
    public void testMarchToApril() throws Throwable {
        NextDate nextDate0 = new NextDate(m: 3, d: 31, y: 2001);
    }
}
```

El...	Class, %	Method, %	Line, %	Branch, %
✓	50% (1/2)	50% (7/14)	32% (33/102)	27% (36/132)
✓	0% (0/1)	0% (0/7)	0% (0/51)	0% (0/66)
✓	100% (1/1)	100% (7/7)	64% (33/51)	54% (36/66)

Tests passed: 4 of 4 tests - 46 ms

The screenshot shows an IDE with the file `GTv12.java` open. The code defines a class `GTv12` with two test methods: `testCreatesNextDate1()` and `testCreatesNextDate0()`. The `testCreatesNextDate1()` method creates a `NextDate` object for August 8, 1851, and expects the next date to be August 9, 1851. The `testCreatesNextDate0()` method creates a `NextDate` object for August 10, 1896, and expects the next date to be August 11, 1896. The coverage report on the right shows 50% class coverage, 50% method coverage, 49% line coverage, and 48% branch coverage. The test runner at the bottom indicates that 27 out of 27 tests passed in 42 ms.

```
package ee.ut.cs.swt.nextdate;

import ...

public class GTv12 {

    @Test(timeout = 4000)
    public void testCreatesNextDate1() throws Throwable {
        NextDate nextDate0 = new NextDate(m: 1851, d: 1851, y: 1851);
        String string0 = nextDate0.run(month: 8, day: 8, year: 1851);
        assertEquals(expected: "8/9/1851", string0);
    }

    @Test(timeout = 4000)
    public void testCreatesNextDate0() throws Throwable {
        NextDate nextDate0 = new NextDate(m: 1896, d: 1896, y: 1896);
        String string0 = nextDate0.run(month: 10, day: 10, year: 1896);
        assertEquals(expected: "10/11/1896", string0);
    }
}
```

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

Tests passed: 27 of 27 tests - 42 ms

Now the test works again. Nothing surprising.