Java class/method name being tested: isValid			
Test Case #	Requirement	Test Description and Input Data	Expected Output/Result
Test Case #1	The method must prevent invalid dates such as 31 days in a month with 30 or less days.	Create an instance of Date with valid month and year but invalid day. Test data: “test data: “3/39/3022”	False/False
Test Case #2	Number of days in February for a non-leap year shall be 28.	Create an instance of Date with the month = 2, day > 28, and the year is a non-leap year Test Data: “02/29/2022”	False/False
Test Case #3	The method must prevent invalid months from being entered in keeping the months STRICTLY between 1-12.	Create an instance of Date with valid date and year but invalid month. Test Data: “35/05/2022”	False/False
Test Case #4	The method must prevent a month from being entered in as all zeros.	Create an instance of Date with invalid month but valid date and year but invalid day. Test Dates: “00/09/2000”	False/False
Test Case #5	The method must prevent a date from being entered in as all zeros.	Create an instance of Date with Invalid day but valid month and year but invalid day. Test: “06/00/2000”	False/False
Test Case #6	The method must prevent a year from being entered in as all zeros.	Create an instance of Date with Invalid day but valid month and year but invalid day. Test: “06/15/0000”	False/False
Test Case #7	The method must prevent day, month and year from being entered in as all zeros.	Create an instance of Date with All zeros being imputed in days, month, year. Test: “00/00/0000”	False/False

<table>
<tr><td colspan="4">Java class/method name being tested: compareTo</td></tr>
<tr><th>Test Case #</th><th>Requirement</th><th>Test Description and Input Data</th><th>Expected Output/Result</th></tr>
<tr><td>Test Case #1</td><td>Comparing the referenced name to another name that is slightly alphabetically later should return 1.</td><td>Create instances where two first names that are very alphabetically similar but with the same last name and pass both through the method.
Test Data: “Abigail Smith” “Abby Smith”</td><td>1</td></tr>
<tr><td>Test Case #2</td><td>Comparing the referenced name to another name that is slightly alphabetically earlier name should return -1.</td><td>Create instances where two first names that are very alphabetically similar but with the same last name and pass both through the method.
Test Data: “Joseph Smith” “Joshua Smith”</td><td>-1</td></tr>
<tr><td>Test Case #3</td><td>Comparing the referenced name to another name that is equal name should return 0.</td><td>Create instances where two names that are identical and pass both through the method.
Test Data: “Jonathan Chen” “Jonathan Chen”</td><td>0</td></tr>
<tr><td>Test Case #4</td><td>Comparing the referenced name to another name that is alphabetically later.</td><td>Create instances where two unique last names and pass both through the method.
Test Data: “Daniel Wong” “Brian Smith”</td><td>1</td></tr>
<tr><td>Test Case #5</td><td>Comparing the referenced last name to another last name that is alphabetically earlier.</td><td>Create instances where two unique last names and pass both through the method.
Test Data: “Brian Smith” “Daniel Wong”</td><td>-1</td></tr>
</table>

Notes:

TestBed Main -