Java Class Name: `Date` **Method Signature:** `public boolean isValid()`			
Test Case #	**Functional Requirement, or Test Objective**	**Test Description and Input Data**	**Expected result/output**
1	The number of days in February for a non-leap year is 28, and 29 for a leap year. The method shall return false if the date given has 30 days in February	• Create an instance of Date with a valid month and year but with an invalid day, day > 28, and the year is a non-leap year • test input: "2/29/2023"	false
2	The number of days in August is at most, 31 days. The method shall return false if the date is 32 days in August.	• Create an instance of Date with a valid month and year but with an invalid day, day > 31, and August has at most 31 days • test input: "8/32/2021"	false
3	The valid range for the month is month >= 1 and month <= 12. The method shall return false for a month value outside the valid range	• Create an instance of Date with a valid day and year but with an invalid month, month > 12, and the last month of the year is 12 • test input: "23/29/1900"	false
4	The valid range for days in any month must be positive. The method shall return false for a day outside the valid range, such as a negative number.	• Create an instance of Date with a valid month and year but with an invalid day, day < 1, and the first day of a month is always 1 • test input: "9/-30/2000"	false
5	The month, day, and year have to be within a valid range	• Create an instance of Date with a valid day,	true

	month >= 1 and month <= 12 day >= 1 and day <= max days for that month. The method shall return true because the month, day, and year are within the valid range	month, and year ● test input: "8/23/2008"	
6	The last day of February during a leap year is 29 instead of 28. The month, day, and year also have to be within a valid range month >= 1 and month <= 12 day >= 1 and day <= 29 for that month. The method shall return true because the month, day, and year are within the valid range.	● Create an instance of Date with a valid day, month, and year ● test input: "2/29/2024"	true

Java Class Name: `Profile` Method Signature: `public int compareTo(Profile other)`			
Test Case #	**Functional Requirement, or Test Objective**	**Test Description and Input Data**	**Expected result/output**
1	The first and second profiles have different first names but the same last names and dates of birth. The method shall return -1 because the name AliceSmith < BobSmith.	• Create an instance of Date and Profile with the same Date and last name and a different first name • test input: profile1: ("Alice", "Smith", "1/1/2000") profile2: ("Bob", "Smith", "1/1/2000")	-1
2	The first and second profiles have different last names but the same first names and dates of birth. The method shall return -1 because the name AliceBrown < AliceSmith.	• Create an instance of Date and Profile with the same Date and first name and a different last name • test input: profile1: ("Alice", "Brown", "1/1/2000") profile2: ("Alice", "Smith", "1/1/2000")	-1
3	The first and second profiles have the same names but different dates of birth. The method shall return -1 because the date 1/1/1990 is before 1/1/2000.	• Create an instance of Date and Profile with the same names but different dates of birth • test input: profile1: ("Alice", "Smith", "1/1/1990") profile2: ("Alice", "Smith", "1/1/2000")	-1
4	The first and second profiles have different first names but the same last names and dates of birth. The method shall return 1 because the name BobSmith > AliceSmith.	• Create an instance of Date and Profile with the same Date and last name and a different first name • test input: profile1: ("Bob", "Smith", "1/1/2000") profile2: ("Alice", "Smith", "1/1/2000")	1

5	The first and second profiles have different last names but the same first names and dates of birth. The method shall return 1 because the name AliceSmith > AliceBrown.	• Create an instance of Date and Profile with the same Date and first name and a different last name • test input: profile1: ("Alice", "Smith", "1/1/2000") profile2: ("Alice", "Brown", "1/1/2000")	1
6	The first and second profiles have the same names but different dates of birth. The method shall return 1 because the date 1/1/2000 is before 1/1/1990.	• Create an instance of Date and Profile with the same names but different dates of birth • test input: profile1: ("Alice", "Smith", "1/1/2000") profile2: ("Alice", "Smith", "1/1/1990")	1
7	The first and second profiles have the same names and dates of birth. The method shall return 0 because AliceSmith == AliceSmith and 1/1/2000 == 1/1/2000	• Create an instance of Date and Profile with the same names and dates of birth • test input: profile1: ("Alice", "Smith", "1/1/2000") profile2A: ("Alice", "Smith", "1/1/2000")	0