| Class Name: Date method: public boolean isValid() //check if a given date is a valid calendar date |
| --- |

| Test Case # | Requirement | Test Description/Input | Expected output |
| --- | --- | --- | --- |
| 1 | Method returns false for 29 days in February for a year that is not a leap year. | Create a Date instance in February with 29 days that isn’t a leap year  Test data: “2/29/2003” | false |
| 2 | Method returns false when given 31 days for months such as April, June, etc. that only have 30 days. | Create a Date instance with 31 days in a month with only 30 days  Test data: “4/31/2003” | false |
| 3 | Method returns false when given a month value higher than 12. | Create a Date instance with a month index that is too high  Test data: “13/31/2003” | false |
| 4 | Method returns false when given a number of days in a month higher than 31. | Create a Date instance with 32 days in the month  Test data: “3/32/2003” | false |
| 5 | Method returns false when provided with months indexed at less than 1, since 1 is January. | Create a Date instance with a month index that is too low  Test data: “-1/31/2003” | false |
| 6 | Method returns true for any date passed through with month indexes 1-12, days 1-31 depending on the month, and a year greater than or equal to 1900 | Create a Date instance with a normal index for month, day, and year  Test data: “4/3/2003” | true |
| 7 | Method returns true for any date passed through with month indexes 1-12, days 1-31 depending on the month, and a year greater than or equal to 1900 | Create a Date instance with a normal index for month, day, and year  Test data: “1/20/2003” | true |

—-----------------------------------------------------------------------------------------------------------------

| Class Name: Student method: public int compareTo(Student st) //compare Student objects to decide what order they should be placed in the roster |
| --- |

| Test Case # | Requirement | Test Description/Input | Expected Output |
| --- | --- | --- | --- |
| 1 | Method should prioritize last names and return 1 if the parameter’s last name takes priority over (this) object | Should return 1 if parameterized Object takes priority  s1 = “John Doe 4/3/2003”  s2 = “Roy Brooks 9/8/1999”  s1.compareTo(s2); | 1 |
| 2 | Method should prioritize earlier birthdays rather than later ones | Should return -1 if (this) Object takes priority  s2 = “Roy Brooks 9/8/1999”  s3 = “Roy Brooks 8/8/1999”  s2.compareTo(s3); | -1 |
| 3 | Method should prioritize last names and return 1 if the parameter’s last name takes priority over (this) object | Should return -1 if (this) Object takes priority  s2 = “Roy Brooks 9/8/1999”  s4 = “Kate Lindsey 7/15/2002”  s2.compareTo(s4); | -1 |
| 4 | If last names are equal, method should prioritize first names and return 1 if the parameter’s first name takes priority over (this) object | Should return 1 if parameterized Object takes priority  s1 = “John Doe 4/3/2003”  s5 = “Jane Doe 5/1/1996”  s1.compareTo(s5); | 1 |
| 5 | Method should prioritize earlier birthdays rather than later ones | Should return 1 if parameterized Object takes priority  s2 = “Roy Brooks 9/8/1999”  s3 = “Roy Brooks 8/8/1999”  s3.compareTo(s2); | 1 |