**Project 1 Test Specifications**

Class Name: Date

Method Signature: public boolean isValid() {} //checks if a given date is a valid calendar date

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case # | Requirement | Test Description and Input Data | Expected Result/Output |
| 1 | Number of days in February for a non-leap year should be 28. The method will return false if the date given has 29 days for a non-leap year. | \* Create an instance of date with February with the day being the 29th and being a non-leap year.  \* test data: “2/29/2011” | false |
| 2 | Number of days in February for a leap year should be 29. The method will return true if the date given has 29 days for a non-leap year. | * Create an instance of date with February with the day being the 29th but it is a leap year. * Test data: “2/29/2016” | true |
| 3 | The number of months available should be from 1-12. The method will return false if a month in a date given is not within the range. | * Create an instance of date with an invalid month value outside the range of 1-12. * Test data: “13/21/1999” | false |
| 4 | The number of days should be valid. The method will return false if an invalid number of days is given. | * Create an instance of a date with a day count of 32. * Test data:   “3/32/2003” | false |
| 5 | Any year given below 1900 shall return false. | * Create an instance of a date having a year before 1900 * Test data: “1/29/1899” | false |
| 6 | Some months have 30 days and some have 31 days. The method shall return false if an incorrect number of days for a specified month is given. | * Create an instance of a date having an invalid number of days for a specified month, for example April only has 30 days * Test data: “4/31/2003” | false |
| 7 | A date with a valid number of days in a specified month. This method will return true if the date has a correct number of days for a given month. | * Create an instance of a date having a valid number of days for a specified month. * Test data: “1/31/2003” | false |

Class Name: Student

Method Signature: public int compareTo(Student student) {} //returns positive if profile of the student is lexicographically higher, 0 if same, and negative if the student’s profile is lexicographically lower

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case # | Requirement | Test Description and Input Data | Expected Result/Output |
| 1 | If the student’s last name is lexicographically higher, this method will return a positive number. | * Create two instances of students where the first student’s last name is lexicographically higher. * Test data: Student 1: “Emily Garcia” and Student 2: “Bob Bojan” | 5 |
| 2 | If the student’s last name is lexicographically lower, this method will return a negative number. | * Create two instances of students where the first student’s last name is lexicographically lower * Test data: Student 1: “Bob Bojan” and Student 2: “Emily Garcia” | -5 |
| 3 | If both students have the same last name, compare the first name lexicographically.  If the first student has a first name that is lexicographically higher, it will return a positive number , else negative if the first name is lexicographically lower. | * Create two instances of students where both students’ last names are the same, but have different first names. * Test data:   Student 1: “Abigail Smith” and Student 2: “John Smith” | -9 |
| 4 | If both students have the same last and first name, it will compare the date of birth. If the first student’s dob is more recent, it will return a positive number, else it will return a negative number if the first student’s dob is older than the second student. | * Create two instances of students where both students have the same last and first name but different date of births. * Test data: Student 1: “John Smith, dob: (9/21/1999)”   Student 2: “John Smith, dob: (9/21/1995) | 1 |
| 5 | If both students have the same last and first name, and the same date of birth, then they are lexicographically equal. The method will return 0. | * Create two instances of students where both students have the same last and first name and date of birth. * Test data: Student 1: “John Smith, dob: (9/21/1999)   Student 2: “John Smith, dob: (9/21/1999) | 0 |