Class name: Student
Method signature: public int compareTo(Student obj) //check if a given student the same as the student argument

Test Case #	Requirement	Test Description and Input Data * **Note:** for input, [John Doe] refers to Student with name "John Doe" for shorthand	Expected Result/Output
1	The method should return "-1" in the case that the first student's name is lexicographically "less" than that of the second student. (Test first name comparison)	● Create an instance of Student with name "John Doe" ● Create an instance of Student with name less than "John Doe" ● Input: [Ivy Doe] .compareTo([John Doe])	-1
2	The method should return "1" in the case that the first student's name is lexicographically "greater" than that of the second student. (Test last name comparison)	● Create an instance of Student with name "John Doe" ● Create an instance of Student with name greater than "John Doe" ● Input: [John Evans] .compareTo([John Doe])	1
3	The method should return "0" if the profile, majors, and credits are the same	● Create an instance of Student with name "John Doe" ● Compare to self ● Input: [John Doe] .compareTo([John Doe])	0
4	Comparison of names should not be case sensitive, so two students of the same name with different capitalizations should be considered equal and return "0".	● Create an instance of Student with name "John Doe" ● Create an instance of Student with same name and random capitalization (such as "jOhN DOe") ● Input: [jOhN DOe] .compareTo([John Doe])	0
5	Compare should return "1" if major is lexicographically greater than compared	● Create an instance of Student with Major "BAIT"	1

		● Create an instance of Student with same Profile and Credits, but major “EE” ● Input: [EE].compareTo([BAIT])	
6	Comparison of majors should be case insensitive, so if Profile and credits same but Major capitalization different, should return “0”	● Create an instance of Student with Major “BAIT” ● Create an instance of Student with Major “bAIt” ● Input: [bAIT] .compareTo([BAIT])	0
7	Should return “1” if student has same profile and major but more credits completed	● Create an instance of Student with, say, 10 credits ● Create an instance of Student with >10 credits ● Input: [11].compareTo([10])	1

Class name: Date
Method signature: public boolean isValid() //check if a given date is a valid calendar date

Test Case #	Requirement	Test Description and Input Data	Expected Result/Output
1	This method shall return true for any date prior to today and on the calendar.	● Create any instance of a valid Date ● Input: “10/24/2001”	true
2	The number of days in February for a leap year is 28. This method shall return true if the date given is 29 only during a leap year.	● Create an instance of date with month = 2, day > 28, and leap-year ● Input: “2/29/2004”	true
3	Because we are using date to validate birthdates, it is impossible to exist now and have a birthday in the future. This method shall return false if the date is anyday beyond today.	● Create an instance of date in the future ● Input: “5/18/2030”	false
4	The number of days in February	● Create an instance of Date	false

	for a non-leap year is 28. This method shall return false if the date given has 29 or more days for a non-leap year.	with month = 2, day > 28, and non-leap year ● Input: “2/29/2003”	
5	Only Jan, March, May, July, Aug, Oct, Dec are allowed to have 31 days. This method shall return false if the date given has 30 or more days and is not a listed month.	● Create an instance of Date with day = 31, but month is not a 31 day month ● Input: “4/31/2003”	false
6	The number of months in a year is only valid between 1 and 12. This method shall return false if the month does not fall within this range.	● Create an instance of Date with valid day and year but invalid month (< 1 or > 12) ● Input: “13/31/2003”	false
7	The number of days in a month is only valid between 1 and 31 maximum. This method shall return false if the day does not fall within this range.	● Create an instance of Date with valid month and year but invalid day (< 1 or > 31) ● Input: “3/32/2003”	false
8	The number of months in a year is only valid between 1 and 12. This method shall return false if the month does not fall within this range.	● Create an instance of date with a negative value ● Input: “-1/31/2003”	false