Item	Possible Points	Earned Points	Notes
Program design for Project • The `main()` method: o uses an array to store Student objects. o uses a `for` loop for entry of the student data. o uses the methods provided in the start file to get the information from the user. o sorts the array prior to displaying the information o uses an enhanced `for` loop to display the student information. • The Student class: o stores the last name, first name, and score. o has appropriate constructors. o has get and set methods for each instance variable o implements the IComparable interface so the students can be sorted by last name. If two students have the same last name, the first name is used to determine the sort order. • The results are formatted correctly (last name, comma, space, first name, colon, space, score). This may be done by overriding the `toString` method in the Student class, by providing a specific method such as `getDisplayText` in the Student class, or by using the getters in the `main()` method to retrieve the data from the object and then formatting the data appropriately.	**5**	3	The output is not sorted. In order for this to work, IComparable would need to inherit the Comparable interface. Then the compareTo method would need to return the correct value. See the second page for a correct compareTo method.
Design Diagrams: • A correct class diagram is provided for all classes • Design documentation reflects actual logic of code • All methods are documented (one diagram for each method; you may have more than one diagram on a page) • No diagram is larger than one page (8 ½ by 11 inches with ½ inch margins on all sides) • If using flowcharts to diagram the logic: o Each flowchart begins and ends with a terminator symbol Note: the main method beginning terminator contains the word `main()`. The main method ending terminator contains the word `return`. Because you do not write the code that calls the main method, you will not have any flowcharts where the beginning terminator contains the word `START` and the ending terminator contains the word `END`. o The appropriate symbol is used o Only one task per process symbol (the rectangle); each variable declaration should be in its own symbol; show the entire formula for calculations o Every symbol (except a terminator) has at least one flowline leading to it and one and only one flowline leading from it. • If using structured pseudocode to diagram the logic: o The pseudocode is appropriately indented o Each variable declaration is on its own line o The entire formula is shown for calculations o Selection and iteration blocks have a clear beginning and ending • If using Warnier Diagrams to diagram the logic: o Braces are appropriately labeled o Each variable declaration is on its own line o The entire formula is shown for calculations	**5**	2	No class diagram for the Student class. Many diagrams do not have an ending terminator. The diagram for the compareTo method only has a beginning terminator.

Item	Possible Points	Earned Points	Notes
Following course standards: • Code standards: o Code restricted to 80 columns o Follows naming conventions for classes, variables, methods, and constants o Appropriate comment block at top of program file (may use javadoc conventions) o Methods appropriately commented (may use javadoc conventions) o Variables have meaningful names o Braces align correctly o Control statements formatted correctly • All non-code files contain your name, the course code (CITP 190), and the project number at the top of the file. • All design diagrams are in one file. • All files are in standard 8 ½ by 11 inch format with at least ½ inch margins on all sides of the page.	**5**	5	Okay
Proof • Shows all test data	**3**	1	The proof does not show the correct output which should be sorted by last name and then by first name.
Screen capture(s)	**2**	2	Okay
Penalties: • Incorrect calculations • Output is not presented as shown (including spelling and spacing) • Code does not follow the standards • Not all test data was used • Reflects material outside what has been covered through Chapter 10 • Using any classes not mentioned in the instructions or does not use one of the classes mentioned in the instructions • Using a `continue` statement or misusing a `break` statement	**-20 for any of the items listed**		
Total	**20**	**13.0**	

```
// use the compareTo method of the String class to compare names
public int compareTo(Object obj)
{
	Student s = (Student) obj;
	int studentCompare = this.lastName.compareTo(s.getLastName());
	if (studentCompare != 0)
	{
		return studentCompare;
	}
	return this.firstName.compareTo(s.getFirstName());
}
```