Lab 1

Reminder: Your code is to be designed and written by only you and not to be shared with anyone else. See the Course Outline for details explaining the policies on Academic Integrity. Submissions that violate the Academic Integrity policy will be forwarded directly to the Computer Science Academic Integrity Committee.
All materials provided to you for this work are copyrighted, these and all solutions you create for this work cannot be shared in any form (digital, printed or otherwise). Any violations of this will be investigated and reported to Academic Integrity.

Objectives

- Exposure to testing
- Introduction to Classes and Objects in Java

Checkpoints

- CHECKPOINTS areas are found in each week's lab outline. CHECKPOINTS are places where it might be a good idea to check in with your lab TA before progressing if there are things that are unclear to you.
- You do not need to show the TA your progress at each CHECKPOINT; you are only required to submit the file specified at the end of the lab to receive lab credit.

Part I

1. Download `Lab1Part1Tester.java` and `Lab1.java` to your Lab1 working directory.
2. `Lab1Part1Tester.java` tests `Lab1.java` **BUT** `Lab1.java` has logic errors in it. You will need to fix the errors using the `Lab1Part1Tester` output to help identify them.
    a. Compile and run the `Lab1Part1Tester` program from the directory you downloaded the files to.
    To compile: `javac Lab1Part1Tester.java`
    To run: `java Lab1Part1Tester`
    b. Identify the first test that is failing. Uncomment the print statement before the test to give you extra information on what the method is returning relative to what it should return.
    c. DO NOT CHANGE the tester
    d. Fix the method in `Lab1.java`, recompile and rerun the tester until the test passes.
    e. Repeat steps c and d until all tests pass.

CHECK POINT – get help from your lab TA if you are unable to complete this part.

Part II

1. Download `Lab1Part2Tester.java` to your Lab1 working directory.
2. Create a new class in your editor called `Student`. Save this file as `Student.java` in your Lab1 working directory. Recall the format for an empty class (NOTE: your ClassName will be Student):

```
public ClassName {

}
```

3. Download `Lab1Part2Tester.java` to your Lab1 working directory and compile and run it.
4. Implement and **test** `Student.java` following the documentation provided (UML and constructor and method descriptions provided on the following page)
5. Tips on getting started. Compile and run `Lab1Part2Tester` after every step.
   Refer to class notes if you are forgetting what the syntax looks like at any point.
   a. Add the 2 attributes/fields to `Student.java`
   b. Implement the `Student()` constructor in `Student.java`
   c. Implement the `getSID()` method in `Student.java`
   d. Implement the `getGrade()` method in `Student.java`
   e. Uncomment the test in `testConstructorsAndGetters` in `Lab1Part2Tester`
   f. Compile and run. Fix the code if all the tests do not pass

CHECK POINT – get help from your lab TA if you are unable to complete this part.

6. Continue to implement and write tests for the remaining constructor and methods in **Student.java**

Student
- sID: String - grade: int
+ Student() + Student (String, int) + getSID(): String + setSID(String): void + getGrade(): int + setGrade(int): void + toString(): String + equals(Student): boolean

The following is the documentation for the constructors and methods in the Student class

- Constructor `Student()` should set this Student's sID to an empty string and this Student's grade to -1
- Constructor `Student(String theSID, int theGrade)` should set this Student's sID to the String passed in as a parameter and this Student's grade to the int passed in as a parameter
- Method `getSID()` should take no parameters and return this Student's sID
- Method `setSID(String newSID)` should set this Student's sID to the String passed in as a parameter
- Method `getGrade()` should take no parameters and return this Student's grade
- Method `setGrade(int newGrade)` should set this Student's grade to the int passed in as a parameter
- Method `toString()` should take no parameters and return a single String that has this Student's sId and grade concatenated
- Method `equals(Student other)` should return true if this Student's sID matches the sID of the other Student passed in as a parameter

CHECK POINT – get help from your lab TA if you are unable to complete this part.