Brendan Jang

CS 362

Assignment 4

**Documentation**

**Unit Testing**

In this assignment, I wrote Unit Tests to identify and correct the bugs in classroom\_manager.py – specifically the two classes included in this file: Student and Assignment.

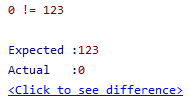
Student class:

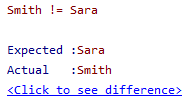
* In test\_student\_constructor(), the code constructs a Student object. It then checks that the id, first\_name, last\_name, and assignments instance variables are set correctly based on the constructor parameters.
* In test\_student\_get\_full\_name(), the code constructs a Student object. It then calls get\_full\_name() to check that the String returned matches the expected result.
* In test\_student\_submit\_assignment(), the code constructs a Student object and an Assignment object. It then calls the submit\_assignment() method and checks that the assignments instance variable is updated correctly.
* In test\_student\_get\_assignments(), the code constructs a Student object and two Assignment objects. It calls the submit\_assignment() method with both assignments and then calls get\_assignments() to check that the assignments are returned properly.
* In test\_student\_get\_assignment\_name(), the code constructs a Student object and two Assignment objects. It calls the submit\_assignment() method with both assignments and then calls get\_assignment(name) to verify that the correct assignment is returned. It also calls get\_assignment(name) on a nonexistent assignment to verify that None is returned.
* In test\_student\_get\_average(), the code constructs a Student object and two Assignment objects, assigning grades to each of them. It calls the submit\_assignment() method with both assignments and then calls get\_average() to verify that the correct average is returned. It then constructs a third assignment with a None grade and calls submit\_assignment() again to verify that the average does not change.
* In test\_student\_remove\_assignment(), the code constructs a Student object and two Assignment objects. It calls remove\_assignment() on both assignment objects and looks at the assignments instance variable to verify that the assignments were removed correctly. The test also tries to remove a nonexistent assignment to verify functionality.

Assignment class:

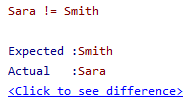
* In test\_assignment\_constructor(), the code constructs an Assignment object. It then checks that the name, max\_score, and grade instance variables are set correctly based on the constructor parameters.
* In test\_assignment\_assign\_grade(), the code constructs an Assignment object. It then calls assign\_grade() to assign a grade to the assignment which is below the max score. It checks that the grade is set correctly. It again calls assign\_grade() to assign a grade to the assignment which is above the max score. It checks that the grade is updated to None.

**Debugging**

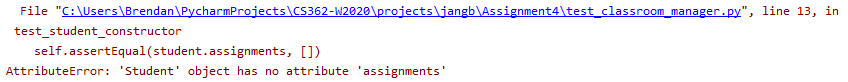
* In test\_student\_constructor(), I found errors in the constructor of the Student class.
  + On line 7, self.id is defaulting to 0 rather than being set to id.
  + On line 8, self.first\_name is being set to last\_name rather than being set to first\_name.



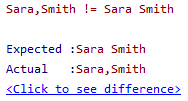
* + On line 9, self.last\_name is being set to first\_name rather than being set to last\_name.



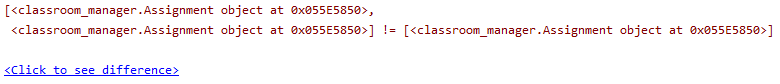
* + On line 10, self.assignmentss is being initialized instead of self.assignments.



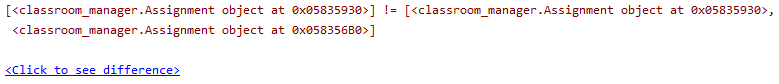
* In test\_student\_get\_full\_name(), I found an error in the String format being returned by the method.
  + On line 13, the code is incorrectly constructing the name as first\_name,last\_name instead of first\_name + “ “ + last\_name.



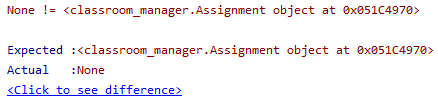
* In test\_student\_submit\_assignment(), I found an error in how assignments are being added to the Student object.
  + On line 17, the code is incorrectly adding the assignment to the assignments instance variable a second time.



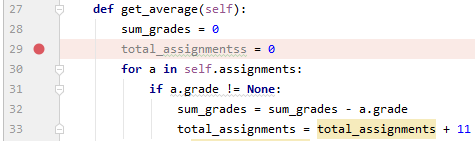
* In test\_student\_get\_assignments(), I found an error in how assignments are being returned by the Student object.
  + On line 20, the code is incorrectly only the first assignment in the list using slicing, which is not needed.



* In test\_student\_get\_assignment\_name(), I found an error in how the assignment is identified to be returned.
  + On line 24, the code is incorrectly checking that the name of the assignment is ‘name’ rather than correctly comparing it to the value of the variable name.



* In test\_student\_get\_average(), I found a few errors in how the average is calculated and then returned.
  + On line 29, the code incorrectly declares total\_assignments as total\_assignmentss.



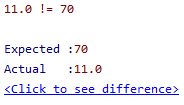
* + On line 32, the code incorrectly updates sum\_grades by subtracting the assignment grade, whereas it should add the assignment grade.



* + On line 33, the code incorrectly increases total\_assignments by 11 for each assignment, instead of the 1 it should increase by.



* + On line 34, average is incorrectly calculated as total\_assignments / sum\_grades, whereas it should be sum\_grades / total\_assignments.

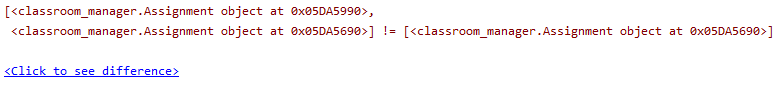


* In test\_student\_remove\_assignment(), I found a few errors in how assignments are removed.
  + On line 39, the code is incorrectly checking that the name of the assignment is ‘name’ rather than correctly comparing it to the value of the variable name.

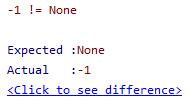
A screenshot of a cell phone

Description automatically generated

* + On line 40, the code is using the wrong operator to remove the entry from the list. Rather than del, it should use list.remove(entry). Here, del name should be replaced by self.assignments.remove(a).



* In test\_assignment\_constructor(), I found an error in the constructor of the Student class.
  + On line 47, the grade instance variable is being initialized incorrectly to -1, but it should be initialized to None.



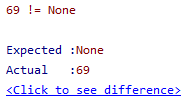
* In test\_assignment\_assign\_grade(), I found several errors in how the grade is assigned to the Assignment object.
  + On line 50, the code mistakenly uses the “==” operator for assignment, instead of the “=” operator.



* + On line 51, the code incorrectly uses the “>=” comparator for checking grades over max\_score, instead of the “>” comparator.



* + On line 52, the code wrongly sets grade to -1 instead of setting self.grade to None in the case where a grade is greater than max\_score.



**Code Coverage**

Running Python code coverage demonstrates that the tests achieve 100% code coverage:

