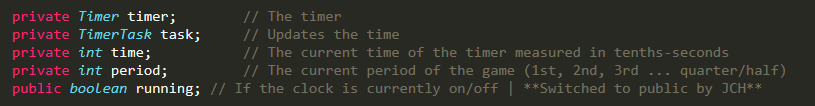
**Project Report Template**

**NOTE: To see my edits refer to my branch -> jc/tests**

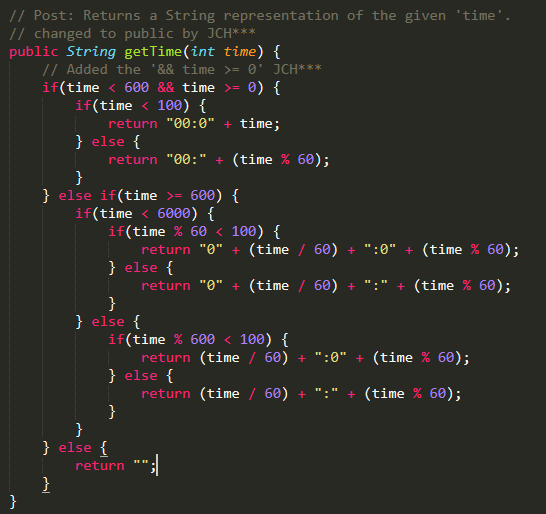
**Unit Testing with Sufficient Coverage**

**GameClock File**

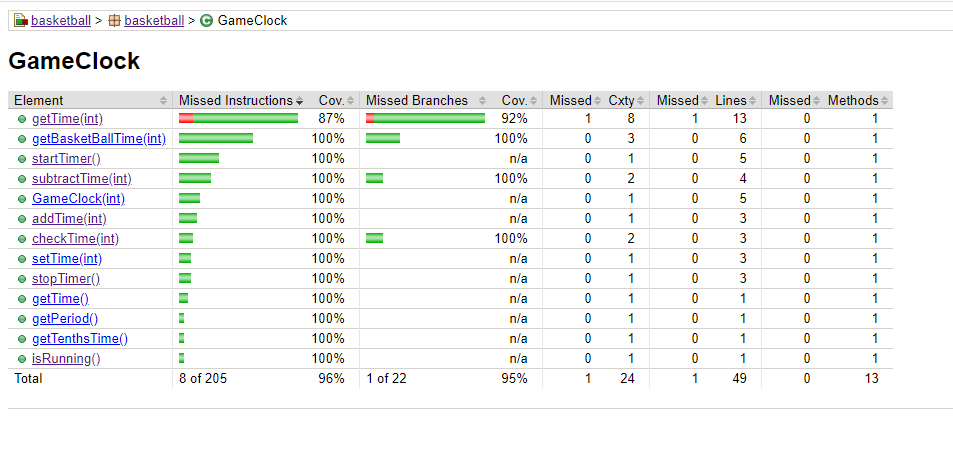
* <https://github.com/csun-comp587-s20/Basketball-Statistics-Tracking/blob/jc/tests/src/test/java/basketball/GameClockTest.java> (File modified)
* Modified and switched certain variables/functions in the main GameClock file to allow me to test specific parameters/aspects (I believe I placed comments by the functions/variables I changed).

****

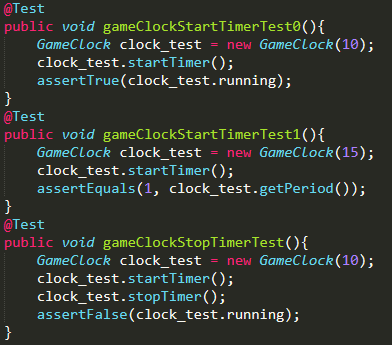
* Modified private String getTime (last function in the GameClock file) to a public function and added a conditional parameter to the if statement -> “&& time >=0.” This conditional parameter was added in order to enable me to reach the final else branch that would essentially return “” due to a different input.

****

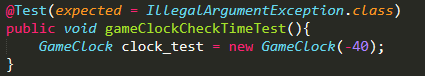
* Original GameClock file for comparison: <https://github.com/csun-comp587-s20/Basketball-Statistics-Tracking/blob/master/src/GameClock.java>
* Tested every function within GameClock file and received 96/95% coverage.

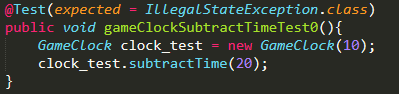


* Swapped out some functions from private to public rather than using Reflection.
* Generated 0-2 test cases for most functions to test various branches and ensure sufficient coverage.
* Simulated Start/Stop timer (spoke to you about this issue but did not have to create helper methods/another interface for it).



* Was able to test exceptions as well (lines 50 and 97 in the first file provided)





* Sufficient coverage (100%) across all functions/elements except for getTime(int) which yielded 87% on missed instructions and 92% on missed branches. There is a line of code (135) in the file that will never be reached regardless of what number is inputted to satisfy the *if* condition. I was not able to figure out a workaround for it.

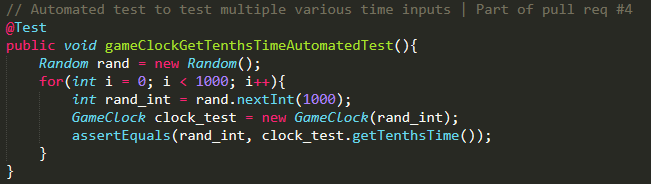
**Player File**

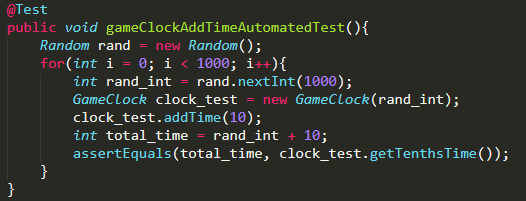
* <https://github.com/csun-comp587-s20/Basketball-Statistics-Tracking/blob/jc/tests/src/test/java/basketball/PlayerTest.java> (File modified)
* Player main file was not modified or altered <https://github.com/csun-comp587-s20/Basketball-Statistics-Tracking/blob/master/src/Player.java>
* Unit tests were written to cover most functions that did not require simulating the game in order to test
* Wrote multiple tests for each function to ensure 100% coverage and functionality.
* Not all of the functions were tested therefore I was not able to achieve 100% coverage.

**Automated Testing**

**GameClock File**

* Created two automated tests (lines 22 and 67) in the following file: <https://github.com/csun-comp587-s20/Basketball-Statistics-Tracking/blob/jc/tests/src/test/java/basketball/GameClockTest.java>

****

****

* Automated tests worked with no problems and contributed to the overall/branch coverage.

**Player File**

* Did not include any automated testing as I was not sure how to test strings in an automated fashion.

**Lessons Learned**

Honestly, it is tough to find specific open source projects to test because most code can be poorly written unless it is maintained by an organization or a highly knowledgeable individual(s). If I could go back and do it from the beginning, I would devote more time to finding a better project that allows for more testability and contains proper code structure. I would be able to perform testing without having to modify specific aspects of the file or call on external libraries/software (Reflection) to help me implement unit tests.