In the Name of God
Advanced Software Testing CA#1

Razieh Dorehgard SID#810199419

3.

testA:
The main disastrous thing in this one is that it is not technically a test since there is no verification for the behavior executed. Thus, the test NEVER FAILS! To make it way better, one can remove the print line and add an assertion for the result and the number 10.

testC:
In this case, the previous issue stays still. Besides, there is a possibility that the method under test throws an exception but IT IS NOT HANDLED in the test. To make it better, one can add an assertion either for the exception as assertThrows or for the expected result.

testInitialization:
2.
The method body contains some setup steps and there is nothing being tested so the test annotation has to be removed and likably be replaced with @BeforeAll or @BeforeEach.

1.
This one can be called a test at least ans owns a setup phase (considering the improvements above) and is verifying a part of the system's behavior. In order to make it better, one can avoid the hard-coded argument of the MUT and replace it with a variable to make the code more readable and maintainable.

4. There is no actual way to test a multi-threaded code. How to simulate the time slot when all threads converge?? Should we mock the threads? The program behavior becomes non-deterministic due to the existing concurrency and therefore unit tests are not capable of testing a multi-threaded code.