Примечания

1. Задание сформулировано для проекта по курсу БД, как самого недавнего большого проекта для студентов. В качестве объекта тестирования может быть выбран любой многокомпонентный проект, написанный или изученный студентом. (Если проект сильно отличается от того, что подразумевалось в курсах ППО и курсовой работе по БД – согласовать с преподавателем.)
2. Проект и тесты могут быть написаны на любых языках программирования (например можно как взять проект на java, а тесты писать на kotlin, так и всё сделать на python).
3. Не надо пытаться сделать свой фреймворк для юнит тестов – нужно использовать типичные для выбранного языка.

Лабораторная работа № 1

Задание

1. Написать unit-тесты для компонентов доступа к данным и бизнес логики выбранного проекта

Требования

1. Требуемое покрытие тестами: один класс - как минимум один test suite / test class с как минимум с двумя тестами (один позитивный, другой негативный) на каждый из public-методов каждого класса основных компонентов; если проект для тестирования выполнен не в объектном стиле -- то необходимо выделить модули исходя из структуры программы
2. Должны быть представлены тесты на обработку исключений (когда ожидаемым результатов является Exception)
3. Должны быть представлены тесты как в классическом (без mock \ stub) так и в “Лондонском” варианте; допустимо представить один и тот же тест в обоих вариантах для сравнения (в учебных целях, на практике это редко имеет смысл)
4. Должна быть соблюдена структура Arrange-Act-Assert для каждого теста с использованием fixture и остальных классов\методов хелперов
5. Если по какой-то причине метод не может быть вызван один в секции Act, то требуется переписать код класса
6. На приватные методы писать тесты не нужны
7. Должны быть представлены тесты с использованием паттерна Data Builder и Fabric(Object Mother) для генерации объектов для тестов
8. Должен быть настроен запуск тестов из командной строки на основании локальной копии репозитория
9. Должен быть представлен автоматически сгенерированный отчет по результатам выполнения тестов (рекомендуется использовать allure – https://github.com/allure-framework – кроме случаев, когда используемый язык программирования не поддерживается); генерация отчета также должна быть учтена в пункте 8

10. Должен быть предусмотрен запуск тестов в случайном порядке
11. Должен быть предусмотрен запуск тестов в режиме без доступа к интернету (идея в том, что тесты только с mock должны в таком случае проходить успешно)
12. Проверить сколько процессов запускается на прогон всех юнит-тестов: отдельный процесс на все тесты, отдельный процесс на каждый тест-класс / тест / и т.д. – разобраться чем это конфигурируется в выбранном стэке, отразить в документации
13. Тесты должны проходить успешно
14. Защита от регрессии, устойчивость к рефакторингу и легкость поддержки -- базовые принципы, которым стоит следовать