**\*Задание 1.** \**Представьте, что вы работаете над разработкой простого приложения для записной книжки, которое позволяет пользователям добавлять, редактировать и удалять контакты.  
Ваша задача - придумать как можно больше различных тестов (юнит-тесты, интеграционные тесты, сквозные тесты) для этого приложения. Напишите название каждого теста, его тип и краткое описание того, что этот тест проверяет.*

1. Тест "Добавить контакт" (юнит тест) - проверяет корректность работы функции добавления нового контакта в базу данных.

2. Тест "Редактировать контакт" (юнит тест) - проверяет корректность редактирования существующего контакта.

3. Тест "Удалить контакт" (юнит тест) - проверяет возможность удаления контакта из базы данных.

4. Тест "Получить все контакты" (интеграционный тест) - тестирует, корректно ли приложение возвращает все контакты из базы данных.

5. Тест "Найти контакт по введенной строке" (сквозной тест) - ищет контакт в базе данных по заданному запросу и проверяет, возвращает ли приложение корректный результат.

6. Тест на "Валидацию полей" (юнит тест) - гарантирует, что все обязательные поля при добавлении или редактировании контакта заполнены.

7. Тест на "Уникальность имени контакта" (юнит тест) - проверка, что в базе данных нет двух контактов с одинаковым именем.

8. Тест "Восстановить удаленный контакт" (интеграционный тест) - восстанавливает удаленный контакт из базы данных и проверяет его корректное отображение.

**\**Задание 2***. *\*Ниже список тестовых сценариев. Ваша задача - определить тип каждого теста (юнит-тест, интеграционный тест, сквозной тест) и объяснить, почему вы так решили.  
 Проверка того, что функция addContact корректно добавляет новый контакт в список контактов"".  
 ""Проверка того, что при добавлении контакта через пользовательский интерфейс, контакт корректно отображается в списке контактов"".  
 ""Проверка полного цикла работы с контактом: создание контакта, его редактирование и последующее удаление"".*

1. Юнит-тест, т.к. если окружить проверяемый модуль заглушками со стороны пользовательского ввода данных, метода валидации и репозитория, то можно изолированно протестировать данную функцию.
2. Интеграционный тест, т.к. этот тест проверяет, как различные части системы взаимодействуют друг с другом, в данном случае, между пользовательским интерфейсом и внутренним функционалом приложения.
3. Сквозной тест, т.к. такой подход обычно затрагивает все элементы тестируемого приложения, а также проверку интеграции с внешними интерфейсами в случае их использования.