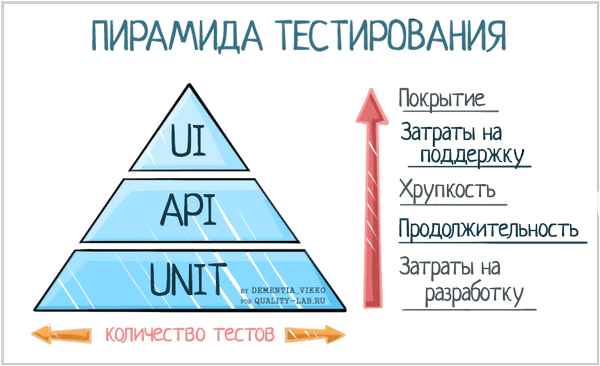
**Пирамида / уровни тестирования**

«**Пирамида тестов**» — абстракция, которая означает группировку тестов программного обеспечения по разным уровням детализации.

Она также даёт представление, сколько тестов должно быть в каждой из этих групп.



Из этой пирамиды главное запомнить два принципа:

1. Писать тесты разной детализации.
2. Чем выше уровень, тем меньше тестов.

Придерживайтесь формы пирамиды, чтобы придумать здоровый, быстрый и поддерживаемый набор тестов.

-Напишите *много* маленьких и быстрых *юнит-тестов*.

-Напишите *несколько* более общих тестов

- и *совсем мало* высокоуровневых сквозных тестов, которые проверяют приложение от начала до конца.

**Подробнее о Классической пирамиде тестирования:**

**Модульные тесты** должны составлять основную часть автоматизированного тестирования.  
- Задачи автоматизации не закрываются до тех пор, пока эти скрипты не будут запущены на реализованной функциональности;  
- Разработка одновременно с модульными тестами заставляет разработчиков задуматься о проблеме, которую они решают, и о любых крайних случаях, с которыми они могут столкнуться;  
- Тесты являются детальными и могут помочь точно определить дефект;  
- Время выполнения невероятно быстрое, потому что им не нужно полагаться на какой-либо пользовательский интерфейс или внешние системы, такие как база данных или API;  
- Они недорогие, просто пишутся, легко поддерживать.

**Интеграционные тесты** должны занимать середину пирамиды.  
Используйте этот уровень для проверки бизнес-логики без использования пользовательского интерфейса (UI);  
Тестируя за пределами пользовательского интерфейса, вы можете тестировать входы и выходы API или сервисов без всех сложностей, которые вводит пользовательский интерфейс;  
Эти тесты медленнее и сложнее, чем модульные тесты, потому что им может потребоваться доступ к базе данных или другим компонентам.

**Тесты пользовательского интерфейса** должны размещаться на вершине пирамиды.  
Большая часть вашего кода и бизнес-логики должна быть уже протестирована до этого уровня;  
Тесты интерфейса пишутся, чтобы убедиться, что сам интерфейс работает правильно;  
Тесты пользовательского интерфейса медленнее и тяжелее в написании и поддержке, поэтому необходимо сводить их к минимуму.

**Перевернутая пирамида тестирования или "Рожок мороженного"**

Перевернутая пирамида считается не рекомендованной, хотя в практике такой подход встречается.

**Основные тезисы:**

1. Тесты пользовательского интерфейса должны автоматизироваться в большей мере.  
2. Длинные тестовые прогоны. Время выполнения занимает намного больше времени, чем другие типы тестов, потому что оно основано на взаимодействии с визуальными элементами пользовательского интерфейса и не обязательно имеет хуки в исходном коде;  
3. Сложно поддерживать, так как тесты пользовательского интерфейса сложно писать и они очень сильно зависят даже от малейших изменений;  
4. Больше подходит для сценариев позитивного пути. Тестирование отрицательных путей в сквозных тестах очень затратно и долго выполняется по сравнению с тестами более низкого уровня;  
5. Ожидание написания модульных тестов до тех пор, пока функции не будут завершены, может привести к тому, что каждому придется несколько раз выполнить большую работу для решения проблемы.

