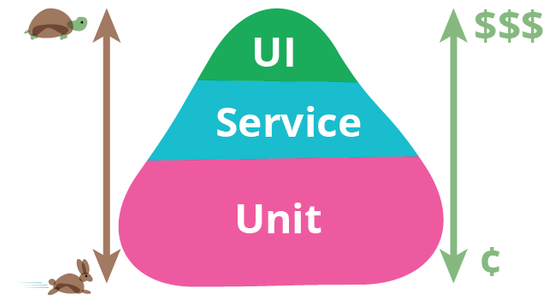
**Юніт тести.**

1. **Юніт тестування**

* **Юніт тестування (англ. unit testing) або модульне тестування —** процес в програмуванні, який дозволяє перевірити на коректність окремі частини коду програми.
* Ідея полягає в тому, щоб писати тести для кожної нетривіальної функції або методу. Це дозволяє досить швидко перевірити, чи не привела чергова зміна в коді до регресії, тобто до появи помилок у вже працюючих, протестованих частинах програми, а також полегшує пошук і усунення таких помилок.

1. **Види тестів**

* **UI тести -** тести на повний продукт, який тестується через взаємодію з користувацьким інтерфейсом. Частіше за все це проводиться вручну, але є фреймворки, які дозволяють автоматизувати цей процес. Наприклад вони можуть емулювати натискання мишкою на кнопки програми, вводити текст в поля вводу і т.і.
* **Інтеграційні тести –** це тести на окремі модулі системи, які вже можуть залежати від зовнішніх систем, тобто файлової системи, мережі, бази даних. Наприклад, це можуть бути тести на функції DLL бібліотеки, тоді при тестуванні відбувається перевірка роботи модуля в цілому, без якихось штучних об’єктів.
* **Юніт тести –** тести на найменші частини програми: функції і класи. Повинні бути ізольовані від зовнішніх систем, і якщо треба використовувати штучні об’єкти, наприклад mock objects.



* Вище зображена збалансована піраміда тестування. Тобто в ідеальному випадку, так повинні співвідноситися за кількістю різні типи тестів в проекті. Бо час виконання і вартість розробки і підтрими тестів збільшується знизу вгору.

1. **Що відрізняє гарні юніт тести**

* не залежать від зовнішніх систем: від мережі, файлової системи і т.і.
* не залежать від інших тестів, порядку їх запуску і т.і.
* тести повинні бути прості. Не повинно бути великих, складних тестів, які перевіряють одразу декілька речей.

1. **Основні конструкції gtest (google test)**

* EXPECT\_EQ, ASSERT\_EQ
* EXPECT\_TRUE, ASSERT\_TRUE
* EXPECT\_THROW, ASSERT\_THROW

1. **Які кейси покривати тестами**

* всі шляхи виконання
* граничні випадки

1. **Що таке TDD**

* Розробка через тестування (TDD) - техніка розробки програмного забезпечення, яка базується на повторенні дуже коротких циклів розробки: спочатку пишеться тест для бажаної зміни в коді, потім пишеться код, який дозволяє пройти тест, і в кінці, якщо треба, проводиться рефакторинг. Розробка через тестування сприяє спрощенню дизайну і всиляє впевненість в програмі.

1. **Користь юніт теств**

* Надійність програми
* Краще розуміння вимог
* Полегшення супроводу, рефакторингу
* Спрощення дизайну
* Документація коду