**Процессы тестирования при разработке программного обеспечения.**

**Процесс тестирования при разработке программного обеспечения (ПО) состоит из нескольких этапов:**

* Планирование тестирования. На этом этапе определяются цели и задачи тестирования, выбираются методы и инструменты, а также составляется план тестирования.
* Разработка тестовых сценариев и тестовых случаев. Тестовые сценарии описывают последовательность действий, необходимых для проверки определённой функциональности ПО. Тестовые случаи представляют собой более детализированные инструкции для выполнения тестовых сценариев.
* Подготовка тестовых данных. Тестовые данные — это наборы входных значений, которые будут использоваться для проверки корректности работы ПО.
* Выполнение тестов. На этом этапе тестировщик применяет тестовые сценарии, тестовые случаи и тестовые данные для проверки работоспособности ПО.
* Анализ результатов. После выполнения тестов анализируются результаты, и обнаруженные ошибки и недоработки документируются.
* Исправление ошибок и повторное тестирование. Разработчики исправляют обнаруженные ошибки, после чего тестировщики проводят повторное тестирование, чтобы убедиться в их исправлении.

**Процессы тестирования при разработке программного обеспечения играют ключевую роль в обеспечении качества и надежности продуктов. Вот основные этапы и типы тестирования:**

* **Функциональное тестирование (Юнит-тестирование)**: Проверка отдельных компонентов (функций или методов) программы. Это помогает выявлять ошибки на ранних этапах разработки.
* **Интеграционное тестирование**: Проверка взаимодействия между разными модулями или компонентами. Цель — убедиться, что они правильно работают вместе.
* **Системное тестирование:** Тестирование всей системы в целом для проверки её соответствия требованиям. Включает функциональное и нефункциональное тестирование.
* **Приемочное тестирование:** Завершающий этап тестирования, проводимый пользователями или заказчиками для оценки, соответствует ли система их ожиданиям и требованиям.
* **Регрессионное тестирование:** Повторное тестирование системы после внесения изменений или исправлений, чтобы убедиться, что эти изменения не вызвали новых ошибок.
* **Тестирование производительности:** Проверка, как система работает под различными нагрузками. Включает нагрузочное, стрессовое и тестирование стабильности.
* **Тестирование безопасности:** Обеспечение защиты системы от внешних угроз и уязвимостей.

Тестирование может быть ручным или автоматизированным.

**Процессы тестирования в разработке ПО (ПО) зависят от выбранной методологии разработки. Рассмотрим несколько распространенных моделей:**

**Waterfall (каскадная модель):**

Требования: Тестирование начинается после завершения этапа кодирования.

Тестирование системы: Проводится после интеграции всех модулей.

Приемочное тестирование: Заказчик проверяет, соответствует ли ПО его требованиям.

**V-Model (V-образная модель):**

Требования: Каждый этап разработки имеет соответствующий этап тестирования.

Модульное тестирование: Проверка отдельных модулей.

Интеграционное тестирование: Проверка взаимодействия модулей.

Системное тестирование: Проверка всей системы на соответствие требованиям.

Приемочное тестирование: Заказчик проверяет систему.

**Agile (Scrum, Kanban):**

Итеративная разработка: Тестирование проводится в каждой итерации (спринте).

Автоматизированное тестирование: Широко используется для автоматизации регрессионных тестов и быстрого получения обратной связи.

TDD (Test-Driven Development): Тесты пишутся до написания кода.

BDD (Behavior-Driven Development): Тесты описывают поведение системы с точки зрения пользователя.

Непрерывная интеграция (CI) и непрерывная поставка (CD): Автоматизация сборки, тестирования и развертывания ПО.

\_\_\_\_\_

Процессы тестирования при разработке программного обеспечения (ПО) - это систематические и организованные мероприятия, направленные на обнаружение дефектов (багов) в ПО и проверку соответствия ПО требованиям. Целью тестирования является повышение качества, надежности и безопасности ПО.

**Вот обзор основных аспектов процессов тестирования:**

**1. Жизненный цикл тестирования (STLC - Software Testing Life Cycle):**

STLC определяет последовательность этапов, которые проходят тестирование в процессе разработки ПО. Хотя конкретные этапы могут варьироваться в зависимости от методологии разработки (например, Agile, Waterfall), обычно выделяют следующие:

Анализ требований: На этом этапе команда тестирования анализирует требования к ПО, чтобы понять, что необходимо протестировать. Выявляются неясности и противоречия в требованиях. Результатом является понимание целей тестирования и критериев приемки.

Планирование тестирования: Определение стратегии тестирования, целей, области охвата, ресурсов, графика и используемых инструментов. Создается план тестирования (Test Plan), который описывает подход к тестированию, роли и обязанности участников, риски и стратегии их смягчения.

Разработка тестовой документации: Создание тестовых сценариев (Test Cases), тестовых наборов (Test Suites) и тестовых данных (Test Data). Тестовые сценарии описывают конкретные шаги, необходимые для проверки определенной функциональности или аспекта ПО.

Настройка тестового окружения: Создание необходимой инфраструктуры для проведения тестирования, включая аппаратное обеспечение, программное обеспечение, сети и данные. Это может включать установку операционных систем, баз данных, серверов приложений и другого необходимого ПО.

Выполнение тестирования: Запуск тестовых сценариев и фиксация результатов. Тестировщики выполняют разработанные тестовые сценарии и сравнивают фактический результат с ожидаемым. Обнаруженные дефекты регистрируются в системе отслеживания ошибок (Bug Tracking System).

Оценка критериев выхода (Exit Criteria Evaluation): Определение критериев завершения тестирования. Анализ покрытия тестами, количества обнаруженных дефектов, серьезности дефектов и стабильности ПО. При достижении критериев выхода, тестирование этапа считается завершенным.

Закрытие цикла тестирования: Подготовка итогового отчета о тестировании (Test Summary Report), содержащего информацию о ходе тестирования, обнаруженных дефектах, качестве ПО и рекомендациях. Анализ уроков, извлеченных из процесса тестирования, для улучшения будущих проектов.

**2. Уровни тестирования:**

Различные уровни тестирования направлены на проверку разных аспектов ПО. Основные уровни:

Модульное тестирование (Unit Testing): Тестирование отдельных модулей или компонентов кода. Выполняется разработчиками для проверки правильности реализации отдельных функций, методов или классов.

Интеграционное тестирование (Integration Testing): Тестирование взаимодействия между различными модулями или компонентами. Проверяется, как отдельные части системы работают вместе. Существуют различные подходы к интеграционному тестированию, такие как “сверху вниз” и “снизу вверх”.

Системное тестирование (System Testing): Тестирование всей системы в целом. Проверяется соответствие системы требованиям и спецификациям. Выполняется командой тестирования после интеграции всех компонентов.

Приемочное тестирование (Acceptance Testing): Тестирование системы конечными пользователями или заказчиками. Проверяется, соответствует ли система их потребностям и ожиданиям. Успешное прохождение приемочного тестирования является условием для приемки системы в эксплуатацию.

**3. Типы тестирования:**

Разнообразие типов тестирования позволяет проверить ПО с разных точек зрения. Некоторые распространенные типы:

Функциональное тестирование: Проверка соответствия функциональности ПО требованиям.

Дымное тестирование (Smoke Testing): Быстрое поверхностное тестирование для проверки работоспособности основных функций.

Регрессионное тестирование (Regression Testing): Повторное тестирование после внесения изменений в код для проверки, что новые изменения не привели к появлению новых дефектов или ухудшению существующей функциональности.

Позитивное тестирование (Positive Testing): Тестирование с использованием корректных данных и сценариев.

Негативное тестирование (Negative Testing): Тестирование с использованием некорректных данных и сценариев для проверки обработки ошибок и исключений.

Нефункциональное тестирование: Проверка нефункциональных характеристик ПО, таких как производительность, безопасность, удобство использования и надежность.

Тестирование производительности (Performance Testing): Оценка скорости, масштабируемости и стабильности системы при различных нагрузках. Включает нагрузочное тестирование, стресс-тестирование и тестирование на выносливость.

Тестирование безопасности (Security Testing): Проверка системы на уязвимости и защита от несанкционированного доступа. Включает тестирование на проникновение, сканирование уязвимостей и аудит кода.

Тестирование юзабилити (Usability Testing): Оценка удобства использования системы для конечных пользователей. Проводится с привлечением реальных пользователей.

Тестирование локализации (Localization Testing): Проверка адаптации ПО для различных языков и регионов.

Тестирование доступности (Accessibility Testing): Проверка доступности ПО для людей с ограниченными возможностями.

Тестирование пользовательского интерфейса (UI Testing): Проверка визуальных элементов и интерактивности интерфейса.

Альфа-тестирование (Alpha Testing): Внутреннее тестирование продукта перед выпуском для ограниченного круга пользователей или тестировщиков.

Бета-тестирование (Beta Testing): Тестирование продукта реальными пользователями в реальной среде перед официальным выпуском.

**4. Подходы к тестированию:**

Ручное тестирование (Manual Testing): Тестирование, выполняемое тестировщиками вручную, без использования автоматизированных инструментов. Используется для исследования нового функционала, тестирования юзабилити и в случаях, когда автоматизация затруднена.

Автоматизированное тестирование (Automated Testing): Тестирование, выполняемое с использованием автоматизированных инструментов и скриптов. Используется для регрессионного тестирования, нагрузочного тестирования и других повторяющихся задач.

**5. Методологии тестирования:**

Waterfall: Последовательный подход, при котором тестирование начинается после завершения разработки.

Agile: Итеративный подход, при котором тестирование выполняется на каждой итерации разработки. Тестировщики работают в тесном сотрудничестве с разработчиками и другими членами команды.

V-Model: Расширение Waterfall, которое связывает каждый этап разработки с соответствующим этапом тестирования.

**6. Инструменты тестирования:**

Существует множество инструментов, облегчающих процесс тестирования:

Системы управления тестами (Test Management Tools): Jira, TestRail, Zephyr.

Инструменты автоматизации тестирования: Selenium, JUnit, TestNG, Cypress, Playwright.

Инструменты управления дефектами (Bug Tracking Tools): Jira, Bugzilla, Mantis.

Инструменты тестирования производительности: JMeter, LoadRunner.

Инструменты тестирования безопасности: OWASP ZAP, Burp Suite.

\_\_\_\_\_

Процессы тестирования при разработке программного обеспечения.

* Планирование тестирования
* Разработка тестовых случаев
* Выполнение тестирования
* Анализ результатов тестирования
* Регрессионное тестирование
* Управление ошибками
* Формирование отчетов
* Документирование

**Разработать модульные тесты для программного модуля Equation, содержащий метод решения квадратного уравнения. Код для программного модуля находится «Resorce/ModuleC#/». Оформить тест-кейсы для модульного тестирования.**