**Регрессионное тестирование: особенности и виды регрессионного тестирования, методы отбора тестов, оценка эффективности.**

**Регрессионное тестирование** — проверка программного обеспечения для подтверждения того, что недавние корректировки софта или кода не сказались негативно на функциональности приложения.

Особенности регрессионного тестирования:

помогает уменьшить количество ошибок после обновлений в программе;

помогает оценить влияние нового кода;

ускоряет вывод продукта на рынок за счёт автоматизации регрессионных тестов;

удешевляет подготовку продукта к запуску: не придётся тратить ресурсы разработчиков на исправление ошибок после релиза.

**Виды регрессионного тестирования:**

Корректирующее. Используют, когда программу обновляют, но не меняют её основные функции. Например, в приложении улучшили дизайн, но не изменили способ оформления заказа.

Прогрессивное. Нужно, когда изменения в программе затрагивают её основные функции. Например, в приложении решили сделать новую функцию — добавлять товары в избранное. Чтобы убедиться, что новая функция не ломает другие процессы, нужно создать новые тесты.

Повторное. Запускают заново после каждого обновления программы. Например, в приложение для обработки платежей внесли изменения, чтобы улучшить процесс оплаты. Тестировщик запускает тесты всей функциональности, чтобы убедиться — новый процесс оплаты не вызвал проблем с другими частями приложения.

Выборочное. Как и прогрессивное, используют, когда изменения в программе затрагивают её основные функции. В случае выборочного создают новые тесты и обновляют те, что есть. Этот метод позволяет выбрать только тесты, которые покрывают изменённые или новые функции программы, чтобы уменьшить время и затраты на тестирование.

Мето**ды отбора тестов:**

Полная регрессия. Тестировщик прогоняет тесты после каждого изменения в программе. Это может занять не день и даже не неделю, но так получится найти максимальное количество ошибок. Метод особенно полезен, когда программу меняют для новой платформы или обновляют операционную систему.

Выбор регрессионного теста. Метод позволяет ограничить регрессионное тестирование только теми частями программы, которые могут затронуть изменения. Это позволяет сосредоточиться на наиболее важных областях и сократить время тестирования. Например, в приложении банка изменили логику авторизации. Можно сосредоточиться только на тестировании этой функции, а транзакции не тестировать.

Приоритизация тест-кейсов. При регрессионном тестировании важно определить порядок запуска тестов. Это помогает сосредоточиться на важных функциях приложения и ускорить процесс проверки. Приоритизация тест-кейсов зависит от частоты возникновения ошибок, задач бизнеса и ожиданий пользователей.

**Оценка эффективности:**

Полнота. Отражает меру отбора тестов, на которых результат выполнения изменённой программы отличен от результата выполнения исходной программы, вследствие чего могут быть обнаружены ошибки.

Точность. Мера способности метода избегать выбора тестов, на которых результат выполнения изменённой программы не будет отличаться от результата её первоначальной версии, то есть тестов, неспособных обнаруживать ошибки.

Эффективность. Оценка вычислительной стоимости стратегии выборочного регрессионного тестирования, то есть стоимости реализации её требований по времени и памяти, а также возможности автоматизации. Относительной эффективностью называется эффективность метода тестирования при условии наличия не более одной ошибки в тестируемой программе. Абсолютной эффективностью называется эффективность метода в реальных условиях, когда оценка количества ошибок в программе не ограничена.

Универсальность. Мера способности метода к применению в достаточно широком диапазоне ситуаций, встречающихся на практике.

**Процент обнаруженных дефектов:**

Определяет количество дефектов, обнаруженных в процессе регрессионного тестирования, по сравнению с общим количеством обнаруженных дефектов.

Высокий процент обнаруженных дефектов указывает на эффективное регрессионное тестирование.

**Покрытие тестирования (Test Coverage):**

Оценка процентного охвата всех функциональных областей системы тест-кейсами.

Повышение покрытия тестирования способствует улучшению качества программного обеспечения и снижению риска незамеченных дефектов.

**Среднее время обнаружения и исправления дефектов:**

Время, затраченное на выявление и исправление дефектов, найденных в процессе регрессионного тестирования.

Снижение среднего времени обнаружения и исправления дефектов указывает на улучшение эффективности процесса.

**Процент автоматизированных тестов:**

Доля автоматизированных тестов от общего количества регрессионных тестов.

Высокий процент автоматизации способствует ускорению тестирования и снижению трудозатрат.

**Количество повторно выявленных дефектов:**

Дефекты, которые были ранее исправлены, но возникли вновь после внесения изменений в код.

Низкое количество повторно выявленных дефектов указывает на высокое качество и стабильность кода.

\_\_\_\_\_

Регрессионное тестирование — это процесс повторного тестирования программного обеспечения после внесения изменений, чтобы убедиться, что изменения не вызвали нежелательных последствий или нарушений в существующей функциональности. Этот вид тестирования является критически важным для поддержания качества и стабильности системы. Давайте рассмотрим его особенности, виды, методы отбора тестов и оценку эффективности.

**Особенности регрессионного тестирования**

Цель: Выявить дефекты, которые могли быть случайно введены в результате изменений кода.

Периодичность: Проводится после каждого изменения или обновления кода.

Автоматизация: Часто автоматизируется для повышения эффективности и уменьшения времени тестирования.

Покрытие: Охватывает как измененные части системы, так и критически важные функции, которые могут быть затронуты изменениями.

**Виды регрессионного тестирования**

Полное регрессионное тестирование: Проверка всей системы целиком, что может быть трудоемким и затратным процессом.

Выборочное регрессионное тестирование: Тестирование только тех частей системы, которые были непосредственно затронуты изменениями.

Инкрементальное регрессионное тестирование: Постепенное добавление новых тестовых случаев к уже существующим, что позволяет сократить время тестирования.

**Методы отбора тестов**

Анализ влияния изменений: Определение того, какие части системы могут быть затронуты изменениями, и отбор соответствующих тестов.

Приоритет тестов: Отбор тестов на основе их важности и критичности для системы.

Покрытие кода: Использование данных о покрытии кода для определения тех участков, которые следует протестировать.

Исторические данные: Анализ предыдущих регрессионных тестов и выявленных дефектов для отбора наиболее важных тестов.

**Оценка эффективности регрессионного тестирования**

Показатели дефектов: Оценка количества и серьезности выявленных дефектов.

Покрытие тестами: Оценка объема покрытия тестами измененных и критически важных участков системы.

Время выполнения тестов: Оценка времени, затраченного на выполнение регрессионных тестов, и их воздействия на процесс разработки.

Стабильность системы: Оценка стабильности системы после внесения изменений и выполнения регрессионных тестов.

\_\_\_\_\_

Регрессионное тестирование - это процесс проверки программного продукта для выявления новых ошибок после внесения изменений в код или добавления нового функционала. Особенностью регрессионного тестирования является то, что его целью не является проверка всех функций программы, а только тех, которые могут быть затронуты в результате изменений. Виды регрессионного тестирования могут варьироваться в зависимости от регрессионных областей (изменения, которые необходимо проверить) и выбранной стратегии тестирования.

Особенности регрессионного тестирования для объектно-ориентированного программирования (ООП) включают проверку внешних и внутренних связей между классами и модулями, проверку изменений в интерфейсах классов и регистрацию нарушений принципов ООП, таких как нарушение инкапсуляции или наследования. Регрессионное тестирование для ООП также требует обновления и адаптации тестового набора для учета изменений в коде, классах и их отношениях, чтобы убедиться, что новые изменения не вызывают регрессионные ошибки.

\_\_\_\_\_

**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.**