**Динамические и статические методы при структурном и объектном подходах тестирования.**

**Динамические методы тестирования**

Динамическое тестирование включает выполнение кода и наблюдение за его поведением во время выполнения. Ключевые виды динамического тестирования включают:

**Функциональное тестирование:**

Проверяет, что система соответствует требованиям и спецификациям.

Включает тестирование на уровне модулей, интеграционное, системное и приемочное тестирование.

**Нефункциональное тестирование:**

Проверяет аспекты, не связанные с функциональностью системы, такие как производительность, безопасность и удобство использования.

Примеры включают нагрузочное тестирование, стресс-тестирование и тестирование на удобство использования.

**Статические методы тестирования**

Статическое тестирование не требует выполнения кода. Оно фокусируется на анализе кода и документации. Основные виды статического тестирования включают:

**Рецензирование (Review):**

Включает анализ кода, проектной документации и требований.

Варианты: формальное рецензирование, неформальное рецензирование, парное программирование.

**Анализ кода:**

Использует автоматизированные инструменты для анализа кода на предмет ошибок, нарушений стиля и потенциальных уязвимостей.

Примеры включают статический анализ кода, анализ потока данных и метрики программного обеспечения.

**Структурный подход к тестированию**

Структурный подход фокусируется на тестировании внутренней структуры системы. Ключевые аспекты:

**Тестирование на основе покрытия кода:**

Покрытие условий, ветвей, путей и операторов.

Гарантирует, что тесты охватывают различные части кода.

**Тестирование потоков данных:**

Анализирует, как данные передаются и обрабатываются внутри системы.

Выявляет ошибки, связанные с неправильным использованием данных.

Объектно-ориентированный подход к тестированию

**Объектно-ориентированный подход учитывает особенности объектно-ориентированного программирования. Основные моменты:**

**Тестирование классов и объектов:**

Проверка методов, атрибутов и взаимодействий объектов.

Использование мок-объектов для изоляции тестируемых классов.

**Тестирование наследования и полиморфизма:**

Проверка корректности наследуемых методов и переопределений.

Тестирование различных реализаций интерфейсов и абстрактных классов.

**Тестирование взаимодействий:**

Анализ взаимодействий между объектами и их зависимостей.

Использование диаграмм последовательностей для идентификации и тестирования взаимодействий.

**\_\_\_\_\_**

**Динамические методы тестирования**

Динамическое тестирование включает выполнение кода программы для проверки его работы и выявления дефектов.

**Структурный подход:**

* Покрытие операторов: Убедиться, что каждый оператор в коде выполняется хотя бы один раз.
* Покрытие ветвей: Проверка всех возможных ветвей условных операторов.
* Покрытие путей: Тестирование всех возможных путей выполнения программы.
* Покрытие условий: Проверка всех логических условий в коде.

**Объектно-ориентированный подход:**

* Тестирование классов и методов: Проверка корректности работы отдельных классов и методов.
* Тестирование взаимодействий: Проверка взаимодействий между объектами.
* Инкапсуляция: Проверка корректного использования инкапсуляции и взаимодействия между компонентами через открытые интерфейсы.
* Статические методы тестирования
* Статическое тестирование проводится без выполнения кода, путем анализа документации, кода и других артефактов.

**Статическое тестирование**

**Структурный подход**:

* Ревью кода: Проверка исходного кода на соответствие стандартам и выявление ошибок.
* Анализ кода (статический анализ): Использование инструментов статического анализа для выявления потенциальных дефектов в коде.
* Инспекции кода: Проверка кода другими разработчиками на предмет ошибок и соответствия стандартам кодирования.
* Проверка соответствия стандартам: Убедиться, что код соответствует установленным стандартам и требованиям.

**Объектно-ориентированный подход:**

* Ревью дизайна классов: Анализ дизайна классов и их взаимодействий для выявления потенциальных проблем.
* Инспекции кода: (Аналогично структурному подходу)
* Проверка UML-диаграмм: Анализ диаграмм классов, последовательностей и других UML-диаграмм для проверки соответствия требованиям.
* Анализ зависимостей: Проверка корректности зависимостей между классами и компонентами.

**Преимущества и недостатки**

**Динамическое тестирование:**

* Преимущества: Обнаружение дефектов во время выполнения программы, проверка реального поведения системы.
* Недостатки: Требует больше времени и ресурсов, зависит от качества тест-кейсов.

**Статическое тестирование:**

* Преимущества: Обнаружение дефектов на ранних стадиях разработки, экономия времени на исправление ошибок.

Недостатки: Не выявляет ошибки, возникающие во время выполнения программы.

\_\_\_\_\_

Статические методы при структурном подходе включают использование инструментов и техник анализа кода без его запуска. Это может включать обзор кода, проверку форматирования и применение линтеров. Статические методы помогают выявить потенциальные проблемы на уровне кода до его выполнения.

Динамические методы при структурном подходе включают запуск программного кода и проверку его поведения во время выполнения. Это может включать тестирование на основе потока управления и потока данных, как описано выше.

При объектном подходе статические методы могут быть использованы для анализа структуры классов и их отношений, например, с помощью анализа UML-диаграмм. Динамические методы могут быть использованы для проверки поведения объектов во время выполнения программы.

\_\_\_\_\_

**Разработать модульные тесты для программного модуля Validation, содержащий метод проверки валидности email. Код для программного модуля находится «Resorce/ModuleC++/». Оформить тест-кейсы для модульного тестирования.**