**Тестирование программного обеспечения (Software Testing)** — это процесс проверки и оценки программного продукта с целью выявления дефектов, ошибок или неправильного функционирования. Оно выполняется для уверенности в том, что программное обеспечение соответствует требованиям, работает корректно и отвечает ожиданиям пользователей.

Цель тестирования программного обеспечения заключается в обеспечении высокого уровня качества продукта, улучшении его надежности и готовности к использованию. Тестирование помогает выявлять ошибки и дефекты на ранних стадиях разработки, что упрощает их исправление и снижает риски возникновения проблем в работе программы после выпуска.

Основные задачи тестирования программного обеспечения включают:

1. Проверку правильности работы программы в соответствии с требованиями и спецификациями.
2. Выявление ошибок, дефектов и несоответствий в функциональности программы.
3. Оценку производительности, надежности и стабильности программы.
4. Проверку совместимости программы с различными операционными системами, браузерами, устройствами и другими факторами окружения.
5. Проверку безопасности и защиты программы от возможных уязвимостей.
6. Установление соответствия программы стандартам и правилам разработки.

Для проведения тестирования программного обеспечения применяются различные методы и подходы. Тестирование программного обеспечения является неотъемлемой частью разработки программных продуктов и помогает повысить их качество, надежность и удовлетворение потребностей пользователей.

**История тестирования программного обеспечения.**

Концепция тестирования программного обеспечения имеет долгую историю, начиная с появления первых компьютеров в середине 20 века. В то время тестирование было в основном связано с проверкой аппаратного обеспечения и исправлением ошибок в электронных схемах. Постепенно с развитием программных систем стала возникать потребность в тестировании программного обеспечения.

В 1947 году Грэйс Хоппер, работавшая над компьютером Mark II, столкнулась с проблемой, когда машина перестала работать из-за ошибки в программе. Хоппер обнаружила, что это было вызвано маленьким насекомым, застрявшим в реле компьютера. Это привело к важному осознанию: программное обеспечение также может содержать ошибки, и его необходимо тестировать.

В 1950-х годах появились первые идеи структурированного подхода к тестированию программного обеспечения. Это включало предварительное планирование, определение тестовых случаев и выполнение тестов с целью проверки правильности работы программы.

В 1960-х годах были разработаны первые модели жизненного цикла программного обеспечения, которые включали этапы тестирования. Например, модель «водопад» предусматривала последовательные фазы разработки, включая фазу тестирования после завершения разработки программы.

В 1970-х годах тестирование программного обеспечения получило большую популярность и было внедрено в процессы разработки программ. В это время появилось понимание того, что тестирование должно быть систематическим и должно охватывать различные аспекты программы.

В 1980-х годах были разработаны методологии и подходы, такие как структурное тестирование, функциональное тестирование и тестирование на основе моделей. В этот период были созданы стандарты и руководства по тестированию, которые помогли систематизировать процесс и повысить его эффективность.

В 1990-х годах с развитием интернета и распространением сложных программных систем возникла необходимость в более сложных методах тестирования.

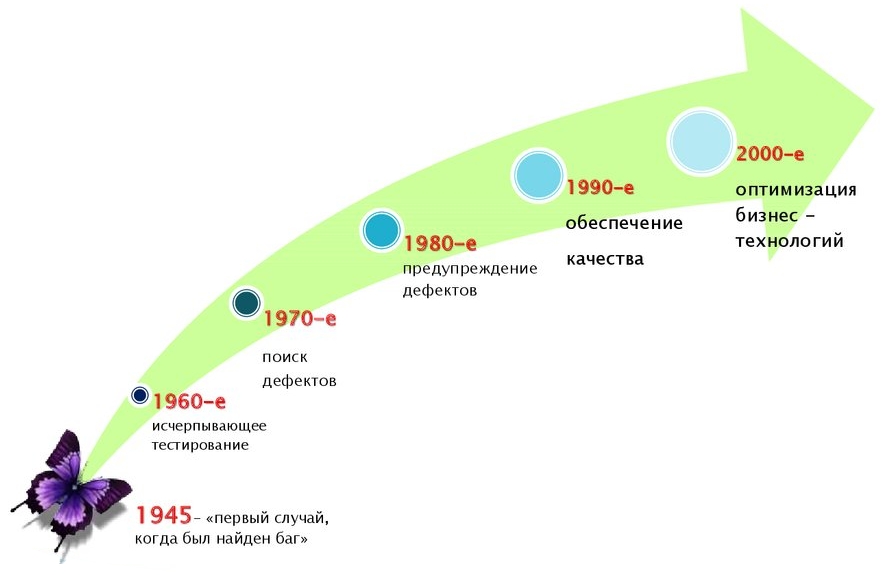
**Методы тестирования программного обеспечения.**

Рисунок . История развития тестирования

Тестирование программного обеспечения можно классифицировать по различным принципам, которые определяют основные аспекты и цели тестирования. Вот некоторые из основных принципов классификации тестирования:

1. По уровню тестирования:
   1. Модульное тестирование: проверка отдельных модулей или компонентов программы.
   2. Интеграционное тестирование: проверка взаимодействия между компонентами системы.
   3. Системное тестирование: проверка системы в целом, включая ее функциональность, производительность и надежность.
   4. Приемочное тестирование: проверка соответствия системы требованиям и ее готовности для принятия в эксплуатацию.
2. По способу проведения тестирования:
   1. Ручное тестирование: тестирование, выполняемое вручную тестировщиками без использования автоматизированных инструментов.
   2. Автоматизированное тестирование: тестирование, выполняемое с использованием специальных инструментов и программных средств для автоматизации выполнения тестовых сценариев.
3. По виду проверяемых атрибутов:
   1. Функциональное тестирование: проверка соответствия функциональным требованиям и ожидаемому поведению системы.
   2. Нефункциональное тестирование: проверка атрибутов качества системы, таких как производительность, надежность, безопасность и удобство использования.
   3. Регрессионное тестирование: проверка, что изменения в системе не привели к появлению новых ошибок или нарушению существующего функционального поведения.
4. По фокусу тестирования:
   1. Белого ящика тестирование: тестирование с полным знанием внутренней структуры и работы системы.
   2. Серого ящика тестирование: тестирование с частичным знанием внутренней структуры и работы системы.
   3. Черного ящика тестирование: тестирование без знания внутренней структуры и работы системы, основанное только на внешнем поведении и требованиях.
5. По времени проведения тестирования:
   1. Тестирование на этапе разработки: тестирование проводится в процессе разработки системы.
   2. Тестирование на этапе внедрения: тестирование проводится перед внедрением системы или ее обновлений в рабочую среду.
   3. Тестирование на этапе эксплуатации: тестирование проводится во время эксплуатации системы для проверки ее работоспособности и стабильности.

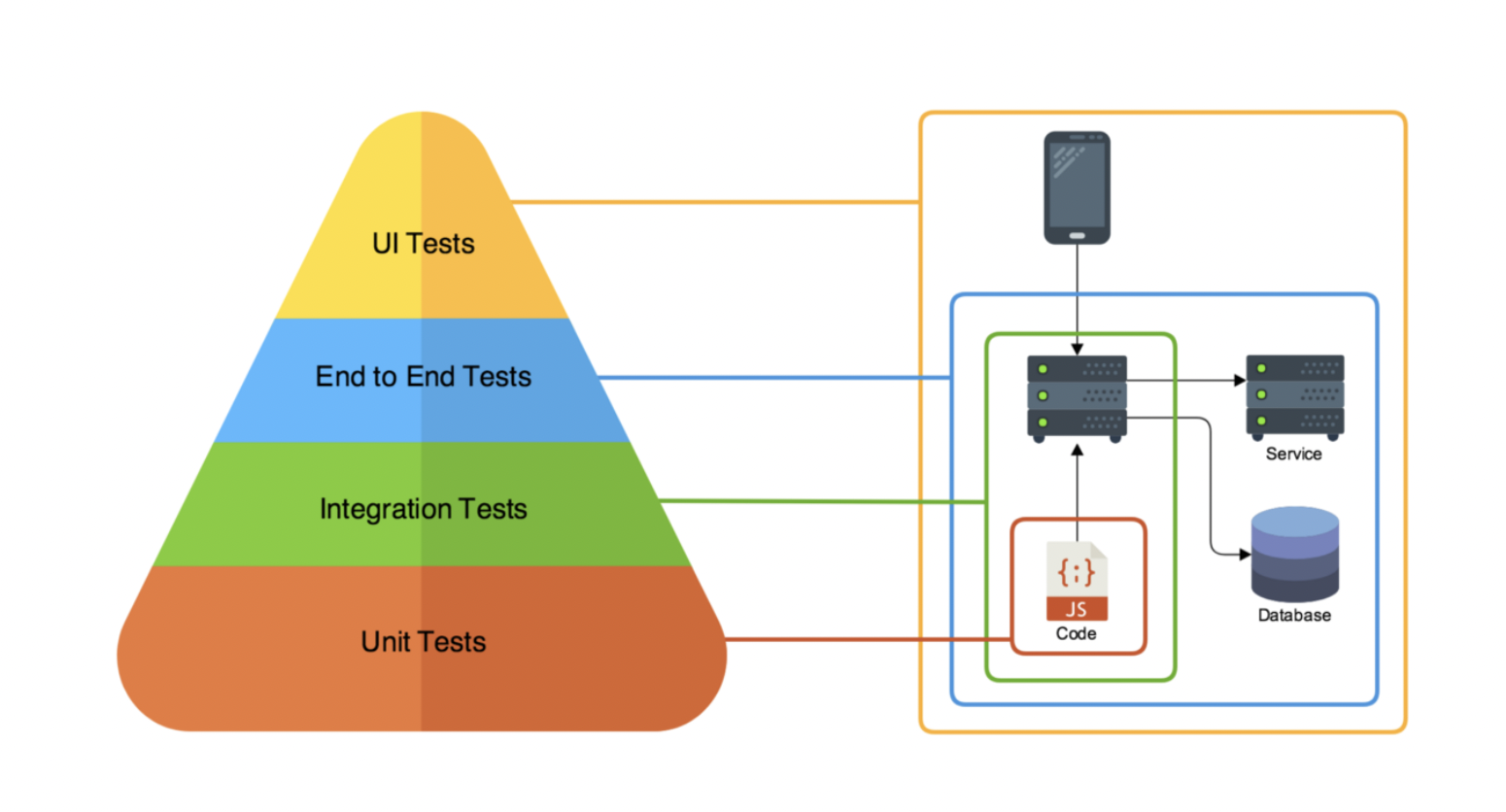
Классификация тестирования по этим принципам помогает определить различные виды и аспекты тестирования, что позволяет более точно организовать и планировать процесс тестирования программного обеспечения.

Рисунок 2. Наглядное изображение уровней тестирования.

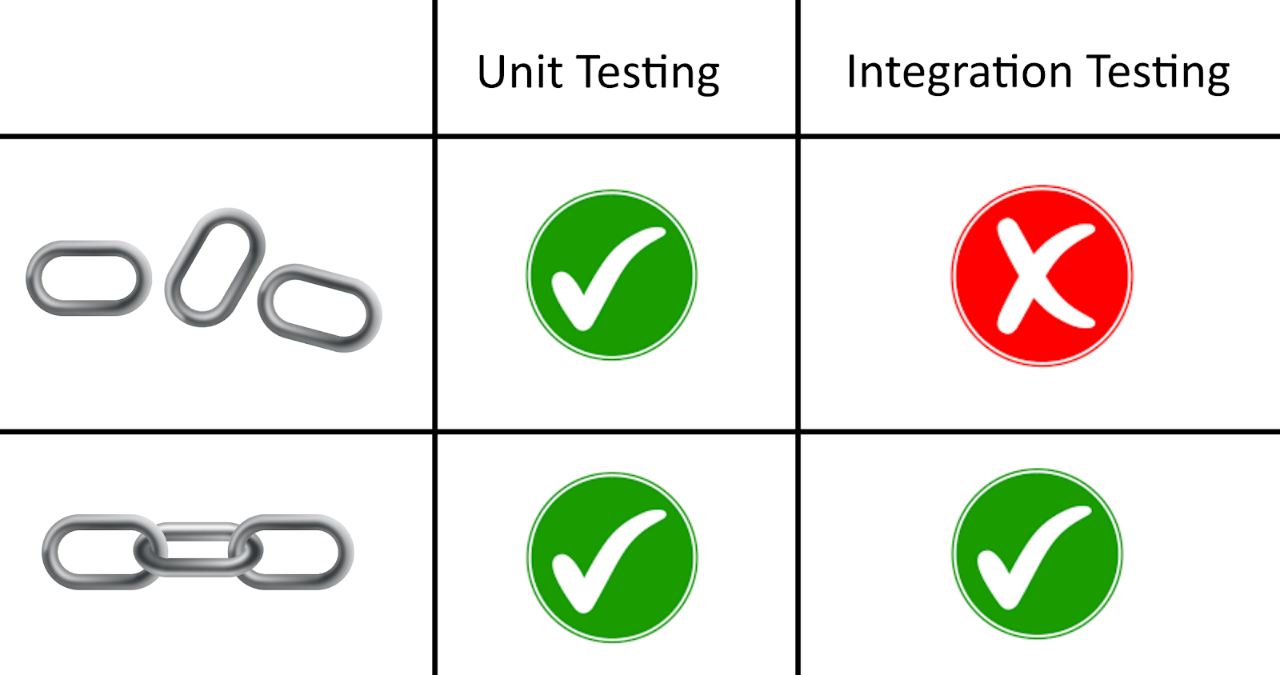
В рамках этого приложения будут рассмотрено функциональное тестирование. А в частности, юнит-тестирование и интеграционное тестирование.

Рисунок . Связь модульного и интеграционного тестирования