# Что такое тестирование программного обеспечения?

- **Тестирование -** процесс исследования ПО с целью получения информации о качестве продукта.

- **Тестирование ПО -** Процесс проверки соответствия заявленных к продукту требований и реально реализованной функциональности, осуществляемый путем наблюдения за его работой в искусственно созданных ситуациях и на ограниченном наборе тестов, выбранных определенным образом.
- **Тестирование** – это один из видов оценки системы с целью найти различия между тем, какой она должна быть, и тем, какая она есть.

# Необходимость тестирования

- Процесс разработки ПО невозможен без контроля качества разрабатываемого продукта;

Тестирование позволяет:

- значительно снизить количество обоснованных претензий к качеству ПО на этапе внедрения;
- сократить ресурсы, задействованные на доработке, исправлении и сопровождении системы;
- затраты на обслуживание ПО уменьшаются на 15-20 %;
- репутация разработчика системы растет.

# Цели и задачи тестирования

- **Цель тестирования** состоит в получении объективной информации о качестве продукта.

– Предоставление информации о качестве ПО конечному заказчику;
– Потввышение качества ПО;
– Предоращение появления дефектов.

- **Задача тестирования -** поиск дефектов.

# Базовая терминология тестирования

**Баг** - это:
-несоответствие между фактическими и требуемыми характеристиками объекта тестирования.
-несоответствие фактического поведения системы разумным ожиданиям пользователя.

- **Тестовые данные** (**test data**) — данные, которые существуют (например, в базе данных) на начало выполнения теста и влияют на работу, или же испытывают влияние со стороны тестируемой системы или компонента.
- Например, для тестирования базы данных нужно создать таблицы, записи; для тестирования - электронного документооборота - создать документы (служебные записки, договора). Иногда нужно создавать данные типа email, логины, пароли, файлы, где надо тестировать разные форматы, размер, разрешение, цветовые параметры и т д.

# Тестовая ситуация -

**это последовательность действий, по которой можно проверить соответствует ли тестируемая функция установленным требованиям.**

Тест-кейс – документ, содержащий набор входных значений, пред- и постусловий, а также ожидаемый результат проведения теста, разработанный для проверки соответствия определенной функциональности системы заданным для этой функциональности требованиям.

- **Отказ** - это:

- симптом, внешнее проявление внутреннего изъяна, наблюдаемое при некоторых условиях.
- тестеры (и пользователи) наталкиваются на отказы/сбои, иначе: наблюдают симптомы.
- изъян кода может не приводить к отказу, т.е. может быть не замечен тестировщиком

Разработчики допускают ошибки при написании кода и в программе затаивается дефект. И даже если дефект не нашли и о нем никто не знает, он все равно есть! Сидит и ждет своего часа. И когда пользователь натыкается на ошибочный код, происходит сбой.

# Кто такой тестировщик?

Тестировщик — это специалист, который занимается тестированием программного обеспечения (ПО) с целью выявления ошибок в его работе и их последующего исправления.

Тестировщик знает, как система работает, где она не работает, и где работает не так, как задумано. Он умеет определить, чем вызвана ошибка, или хотя бы знает, где это искать.

- Работа тестировщика, напоминает работу следователя или детектива!
- Чтобы отыскать спрятавшуюся в глубине программы ошибку, нужно быть изобретательным!

А именно:

- задавать нужные вопросы,
- понимать психологию программиста,
- знать техники тестирования,
- понимать логику работы системы,
- уметь предвидеть, где именно может скрываться баг

и много другое…

# Кто такой QA инженер?

**Quality Assurance engineer** — это специалист по обеспечению качества, деятельность которого направлена на улучшение процесса разработки ПО, предотвращение дефектов и выявление ошибок в работе продукта.
Quality Assurance гарантирует, что процесс поставлен правильно и дает предсказуемый результат.

# Цели и задачи QA инженера

- **Основная задача QA** — обеспечение качества.
- **QA-инженер** фокусирует внимание на процессах разработки ПО, улучшает их, предотвращает появление дефектов и проблем. Предпочитают не лечить, а проводить профилактические мероприятия


Quality Assurance
Quality Control
Тестирование

# Цели и задачи тестировщика и QA инженера

- Тестирование — составляющая часть более весомого понятия Quality Control (контроль качества). QC отвечает за измерение качества продукта, анализ результатов тестирования и качества релизов или сборок продукта, сбор и анализ метрик качества. В свою очередь QC является составной частью Quality Assurance. QA решает более глобальные задачи, главная из которых — управление качеством самого процесса. QA старается предвидеть и предотвратить возможные проблемы (так сказать сработать на опережение). В обязанности QA входит подбор практик, методов, подходов, инструментов и прочее.

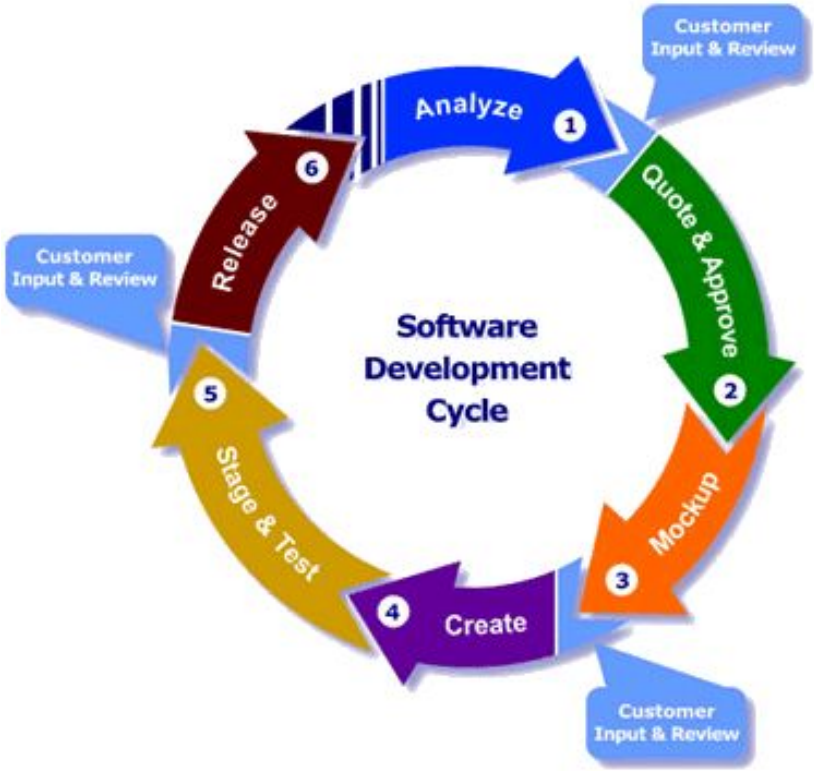
**Тестирование — процесс оценки качества продукта, а QA — это формирование процессов, которые обеспечивают высокое качество ПО (в том числе и процессов разработки, аналитики, документирования).**

# Жизненный цикл ПО

Жизненный цикл программного обеспечения (Software Life Cycle Model) — это период времени, который начинается с момента принятия решения о создании программного продукта и заканчивается в момент его полного изъятия из эксплуатации.

# Обзор стадий разработки программного обеспечения

- Анализ;
- Проектирование;
- Программирование;
- Документирование;
- Тестирование;
- Сопровождение.



**Анализ**

Процесс сбора требований к ПО, их систематизация, документирование, анализ, выявление противоречий и разрешение конфликтов в процессе разработки ПО

**Проектирование**

- Процесс создания ПО;
- Определение внутренних свойств системы и детализация ее внешних свойств на основе требований к ПО.
- Нотации – схематическое выражение характеристик:
  - Блок-схемы;
  - ER-диаграммы;
  - UML-диаграммы;

**Программирование** – процесс создания программ

- Разработка комплекса алгоритмов;
- Написание исходного кода;
- Преобразование в машинный код (компиляция);
- Тестирование и отладка;

**Документация**

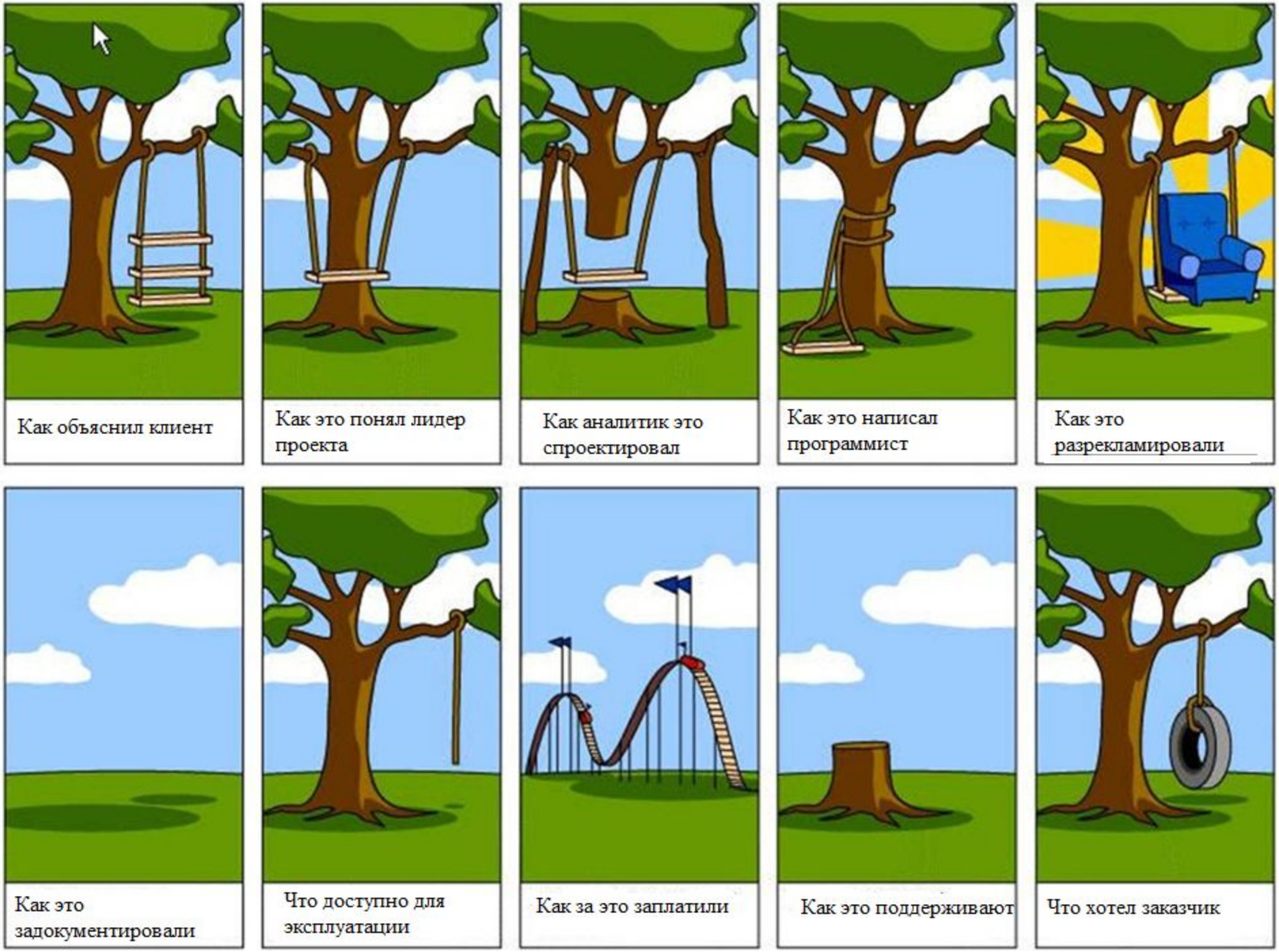
Печатные руководства пользователя, диалоговая документация и справочный текст, описывающий функции и алгоритм использования ПО

Виды:

- Архитектурная/Проектная;
- Техническая;
- Пользовательская;

## Сопровождение

- Сбор и анализ информации от пользователей;
- Создание отчетов об ошибках;
- Требования по выпуску исправлений (hot-fixes, updates, service packs etc) .


Как объяснил клиент
Как это понял лидер проекта
Как аналитик это спроектировал
Как это написал программист
Как это разрекламировали
Как это задокументировали
Что доступно для эксплуатации
Как за это заплатили
Как это поддерживают
Что хотел заказчик

# Обзор моделей разработки

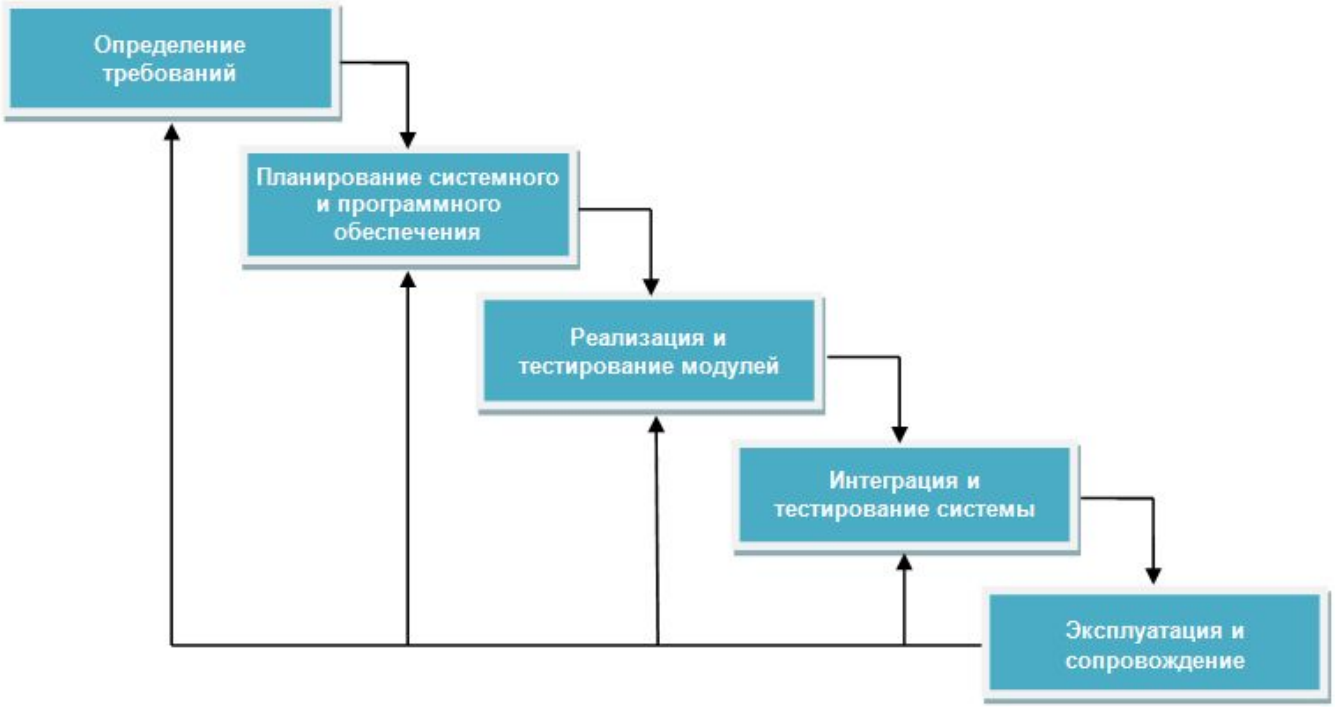
**Виды моделей**

- Водопадная (Каскадная);
- V-модель;
- Спиральная;
- Итерационная.

# Водопадная модель

Последовательность фаз, переходящих от одной к другой;

- Переход к следующей – только после полного завершения предыдущей.



# V-модель

Дальнейшее развитие водопадной модели

Разработка и тестирование идут одновременно;

Цели:

- Минимизация рисков;
- Повышение качества;
- Уменьшение стоимости проекта;
- Коммуникация в команде

# V-образная модель (V-model)


При внесении изменения необходимо выполнить следующую работу:

- ✓ Принять или отклонить изменение.
- ✓ Согласовать изменение с заказчиком.
- ✓ Организовать работы по переделке.

Линия времени

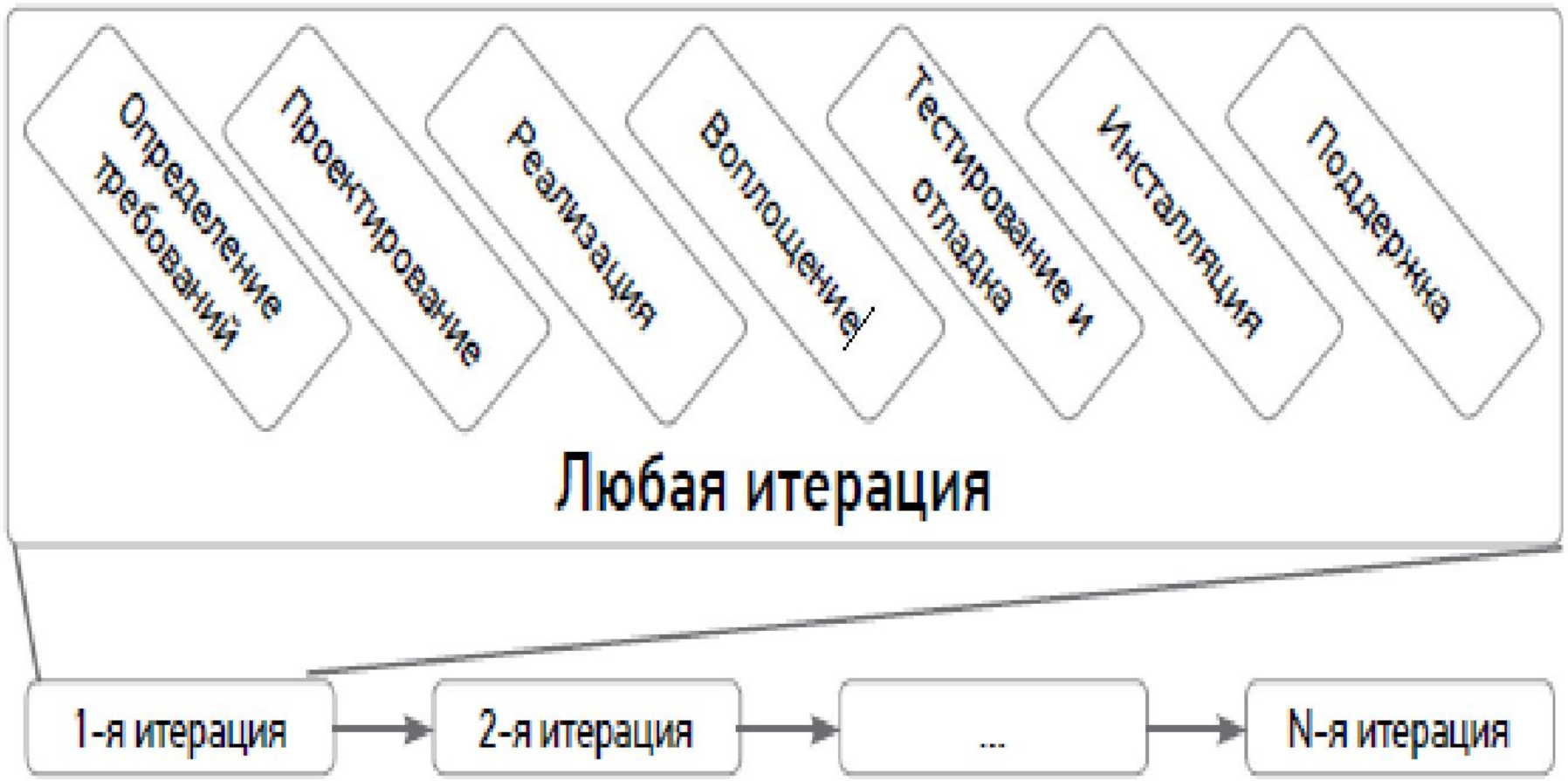
# Спиральная модель

- это модель процесса разработки программного обеспечения, сочетающая в себе проектирование и постадийное прототипирование. Спиральную модель разработки можно схематично представить в виде спирали, значимой единицей которой является виток. Любой виток спирали представляет собой законченный процесс по созданию определенной части продукта, либо выпуску новой версии. Переход между витками осуществляется строго последовательно.
Прототип ПО - это частичная или возможная реализация предлагаемого нового продукта.

# Спиральная модель (Spiral model)

# Итерационная модель

- Выполнение работ параллельно с непрерывным анализом полученных результатов и корректировкой предыдущих этапов работы;
- Планирование-Реализация-Проверка-Оценка (plan-do-check-act).


Определение требований
Проектирование
Реализация
Воплощение
Тестирование и отладка
Инсталляция
Поддержка
Любая итерация
1-я итерация
2-я итерация
...
N-я итерация

# Жизненный цикл тестирования (Testing Life-Cycle)

- анализ требований(Requirements Analysis)
- анализ дизайна проекта (Design Analysis)
- планирование тестирования (Test planning)
- разработка тестов (Test development)
- выполнение тестов (Test Execution)
- написание отчетов (Test Reporting)
- повторная проверка дефектов(Retesting the Defects)

# Планирование тестирования (Test planning)

Для более ясного описания целей и задач тестирования составляется такой документ как тест-план.

**Тест-план – документ, описывающий средства, подходы, график работ и ресурсы, необходимые для проведения тестирования. Помимо прочего, определяет инструменты тестирования, функциональность, которую требуется протестировать, распределение ролей в команде, тестовое окружение, используемые техники тест-дизайна, критерии начала и окончания тестирования и риски.**

## Разработка тестов (Test development)

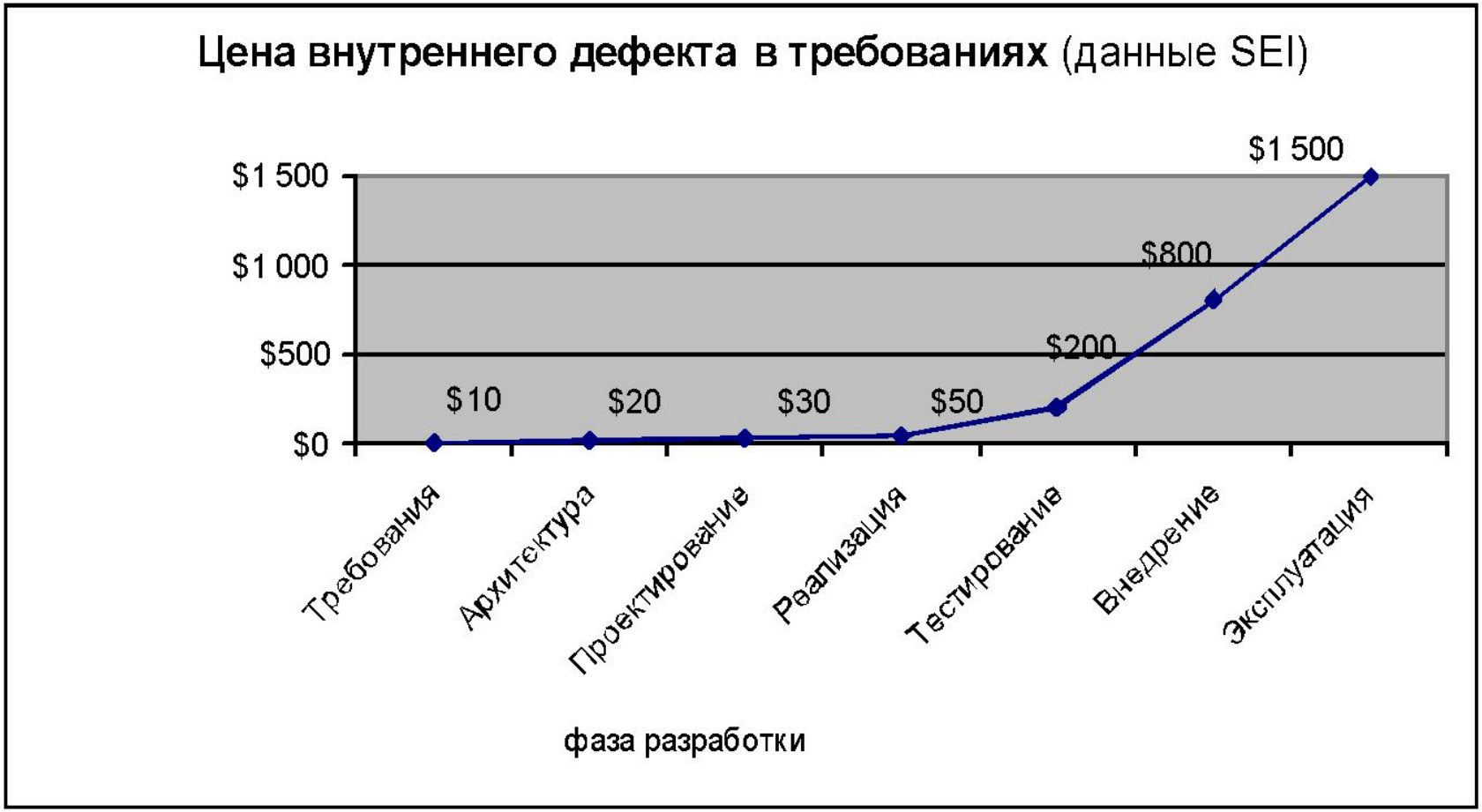
Анализ и проектирование тестов – это процесс написания тестовых сценариев и условий на основе общих целей тестирования.

**Тест-кейс** – документ, содержащий набор входных значений, пред- и постусловий, а также ожидаемый результат проведения теста, разработанный для проверки соответствия определенной функциональности системы заданным для этой функциональности требованиям.

# Написание отчетов (Test Reporting)

После окончания тестирования происходит написание отчета, который будет доступен всем заинтересованным сторонам. Ведь не только тестировщики должны знать результаты выполнения тестов, – эта информация может быть необходима многим участникам процесса создания ПО.

# График стоимости поиска дефекта на различных стадиях разработки проекта



# Exploratory (ознакомительное) и Scripted (по сценарию) тестирование

**Exploratory** - это разработка и выполнения тестов в одно и то же время.

Такое тестирование подразумевает под собой одновременно изучение проекта, функционала, проектирование тест кейсов в уме и тут же их исполнение не записывая и не создавая тестовую документацию.

**Скриптовое тестирование** -тестирование перед началом которого создаются тесты, и уже по ним осуществляются проверки.

# Manual (ручное) и Automated (автоматическое) тестирование

**Manual (ручное)**- проверка системы без использования дополнительных программных средств, ну разве что, Exel, NotPad  и т д..

**Автоматизированное тестирование(автоматическое)** - использование специального программного обеспечения (помимо тестируемого) для контроля выполнения тестов и сравнения ожидаемого и фактического результата работы программы. Этот тип тестирования помогает автоматизировать часто повторяющиеся, но необходимые для максимизации тестового покрытия задачи.

# Тестирование Black Box(черный ящик) и White Box (белый ящик)

| Критерий | Black Box | White Box |
|---|---|---|
| *Определение* | тестирование, как функциональное, так и нефункциональное, не предполагающее знания внутреннего устройства компонента или системы | тестирование, основанное на анализе внутренней структуры компонента или системы |
| *Уровни, к которым применима техника* | В основном:<br>• Приемочное тестирование<br>• Системное тестирование | В основном:<br>• Юнит-тестирование<br>• Интеграционное тестирование |
| *Кто выполняет* | Как правило, тестировщики | Как правило, разработчики |
| *Знание программирования* | Не нужно | Необходимо |
| *Знание реализации* | Не нужно | Необходимо |
| *Основа для тест-кейсов* | Спецификация, требования | Проектная документация |

# Positive (позитивное) и Negative (негативное) тестирование

- «Позитивное» тестирование- проверить результат работы приложения при получении им «правильных» входных данных.
- «Негативное» тестирование - как ведет себя приложение, получая на вход «неправильные» данные.