IT-Academy

ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР ПРОГРАММИРОВАНИЯ И ВЫСОКИХ ТЕХНОЛОГИЙ

Функциональное тестирование ПО

ОСНОВЫ ТЕСТИРОВАНИЯ ПО ВВЕДЕНИЕ

Blockudoku®: block puzzle game
4.4 ★

Art Puzzle - Jigsaw Art Games
4.5 ★

Groovepad - music & beat maker
4.7 ★

Jigsaw Puzzles - puzzle games
4.5 ★

Drum Pad Machine - beat maker
4.4 ★

Number Match - number games
4.4 ★

Pixel Art - Color by Number
4.5 ★

Differences - find & spot them
4.5 ★

Sudoku.com - classic sudoku
4.4 ★

Logic Puzzles - Brain Riddles
4.1 ★

Number Sums - Numbers Game
4.6 ★

Killer Sudoku by Sudoku.com
4.5 ★

Nonogram.com - logic games
4.2 ★

Easy Game - brain test
4.6 ★

Word Search - crossword puzzle
4.7 ★

Backgammon - board game
4.1 ★

Nonogram Color - logic puzzle
4.3 ★

Spider Solitaire
4.5 ★

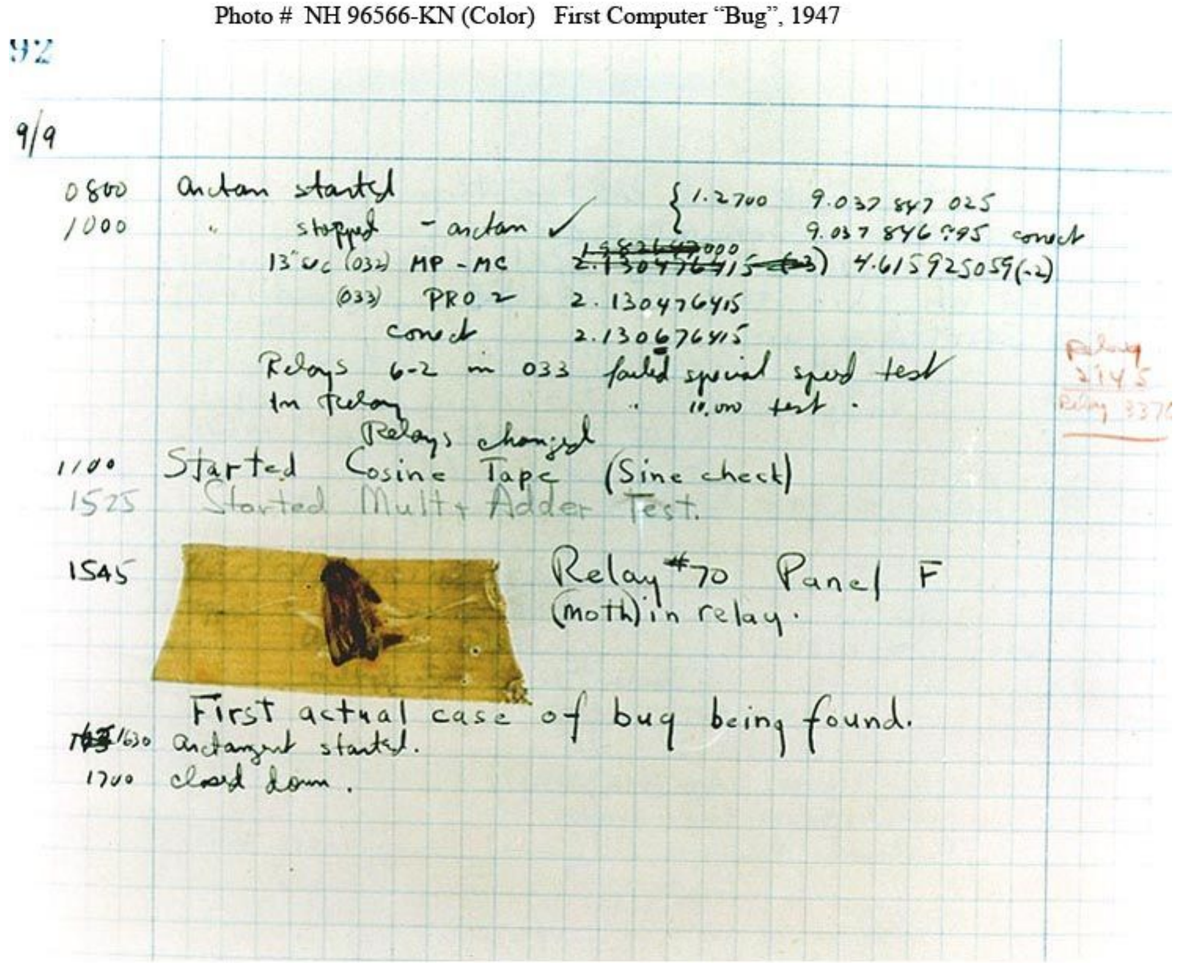
Block Sums - Number Puzzle
4.1 ★

Coindoku - wood block puzzle
4.4 ★

В ЭТОМ РАЗДЕЛЕ:

- НЕМНОГО ИСТОРИИ
- ПСИХОЛОГИЯ ТЕСТИРОВАНИЯ
- ПОЧЕМУ ТЕСТИРОВАНИЕ НЕОБХОДИМО?
- ПРИНЦИПЫ ТЕСТИРОВАНИЯ
- ВВЕДЕНИЕ В ОСНОВНУЮ ТЕРМИНОЛОГИЮ

Когда день тестировщика?



НЕМНОГО ИСТОРИИ

60-е ГОДЫ

В 60-х годах прошлого века основное внимание уделялось т.н. «исчерпывающему тестированию» - проверке всех возможных путей выполнения кода со всеми возможными входными данными.

70-е ГОДЫ

В начале 70-х тестирование ПО обозначалось как «процесс, направленный на демонстрацию корректности продукта» или как «деятельность по подтверждению правильности работы ПО».

80-е ГОДЫ

В 80-х годах тестирование ПО расширилось таким понятием, как предупреждение дефектов.

90-е – 20-е ГОДЫ

В понятие «тестирование» стали включать планирование, проектирование, создание, поддержку и выполнение тестов и тестовых окружений.

СОВРЕМЕННЫЙ ЭТАП

«гибкие методологии, тесная интеграция с разработкой, автоматизация».

ВВЕДЕНИЕ

ПСИХОЛОГИЯ ТЕСТИРОВАНИЯ

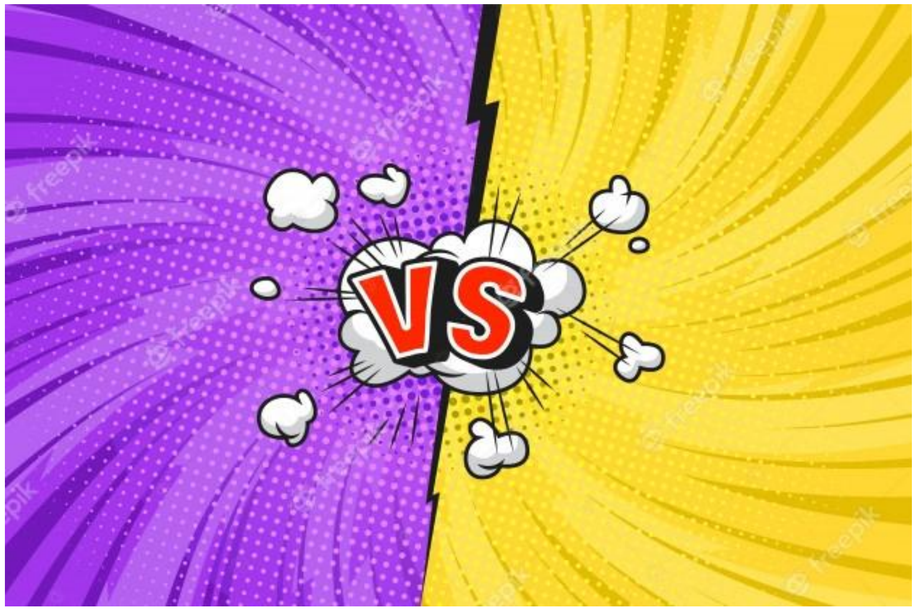
- ХОРОШИЕ КОММУНИКАТИВНЫЕ НАВЫКИ
- СПОСОБНОСТЬ ЯСНО, БЫСТРО, ЧЕТКО ВЫРАЖАТЬ СВОИ МЫСЛИ
- ИСПОЛНИТЕЛЬНОСТЬ
- ОТВЕТСТВЕННОСТЬ
- ТЕРПЕНИЕ,ВНИМАТЕЛЬНОСТЬ К ДЕТАЛЯМ, НАБЛЮДАТЕЛЬНОСТЬ
- ГИБКОЕ МЫШЛЕНИЕ, ХОРОШАЯ СПОСОБНОСТЬ К ОБУЧЕНИЮ
- ХОРОШЕЕ АБСТРАКТНОЕ И АНАЛИТИЧЕСКОЕ МЫШЛЕНИЕ
- СПОСОБНОСТЬ СТАВИТЬ НЕСТАНДАРТНЫЕ ЭКСПЕРИМЕНТЫ
- СКЛОННОСТЬ К ИССЛЕДОВАТЕЛЬСКОЙ ДЕЯТЕЛЬНОСТИ

НАВЫКИ, НЕОБХОДИМЫЕ ДЛЯ QA

1. **АНГЛИЙСКИЙ ЯЗЫК**
2. Не бояться компьютеров и телефонов
3. Базы данных и SQL
4. Понимание принципов работы сетей, операционных систем, приложений
5. Уметь вертеться!!!!
6. Программирование??

ВВЕДЕНИЕ

5 МИФОВ О ТЕСТИРОВАНИИ



- Миф первый: тестирование — это скучно.
- Миф второй: тестирование — это самый простой способ войти в айти.
- Миф третий: тестировщики всего лишь ищут ошибки.
- Миф четвертый: машины заменят тестировщиков, и они станут ненужными.
- Миф пятый: тестировщики не ладят с разработчиками.

ПОЧЕМУ ТЕСТИРОВАНИЕ НЕОБХОДИМО?

БИЗНЕС

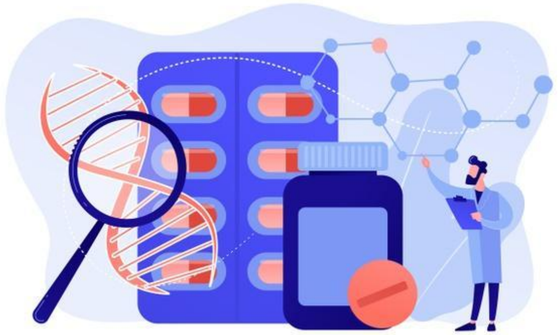
Пользователи склонны пользоваться качественными продуктами (даже если они дороже)

ДАННЫЕ

Пользователи: «лучше не рисковать личными данными, деньгами и т.п.»

БЕЗОПАСНОСТЬ

Все: «Мы не хотим рисковать!»



ВВЕДЕНИЕ

ПОЧЕМУ ТЕСТИРОВАНИЕ НЕОБХОДИМО?

- Растет количество устройств
- Растет количество пользователей
- Растет сложность ПО

- Скорость выхода на рынок является ключевым конкурентным преимуществом сегодня

ВВЕДЕНИЕ

ПОЧЕМУ ТЕСТИРОВАНИЕ НЕОБХОДИМО?

Никто не совершенен!

Чем большее давление на нас оказывается, тем более мы склонны делать ошибки.

В ИТ-разработке мы должны соблюдать временные сроки и бюджет.

Требования определены нечетко или плохо документированы.

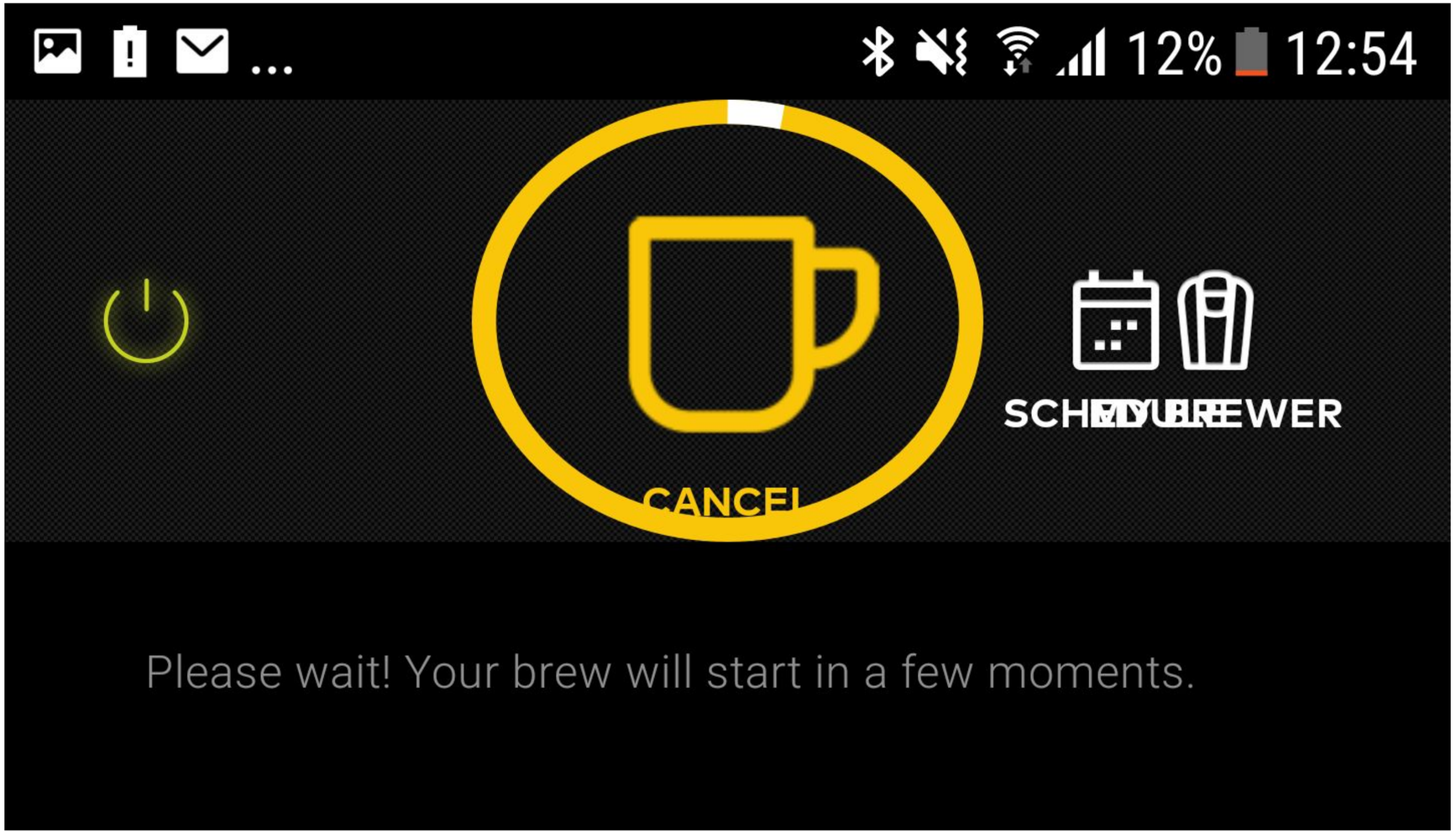
Спецификации данных не завершены.

“

Приведите примеры «багов ПО» из жизни

”

Примеры «багов ПО» из жизни



СЕМЬ ПРИНЦИПОВ ТЕСТИРОВАНИЯ

Принцип 1 – Тестирование демонстрирует наличие дефектов

Тестирование может показать, что дефекты в программном обеспечении есть, но не может доказать, что никаких дефектов нет. Тестирование снижает вероятность того, что в программном обеспечении остались необнаруженные дефекты, но, даже если никаких дефектов не обнаружено, это не доказательство правильности работы программы.

СЕМЬ ПРИНЦИПОВ ТЕСТИРОВАНИЯ

- Принцип 1 – Тестирование демонстрирует наличие дефектов
- Принцип 2 – Исчерпывающее тестирование невозможно

Протестировать абсолютно все (все комбинации входов и предусловий) не представляется возможным, за исключением тривиальных случаев.

Вместо исчерпывающего тестирования, мы используем риски и приоритеты для эффективного сосредоточения усилий тестирования.

СЕМЬ ПРИНЦИПОВ ТЕСТИРОВАНИЯ

- Принцип 1 – Тестирование демонстрирует наличие дефектов
- Принцип 2 – Исчерпывающее тестирование невозможно
- Принцип 3 – Раннее тестирование необходимо

Тестовые активности должны начинаться как можно раньше в цикле разработки программного обеспечении или системы, и должны быть направлены на достижение определенных целей.

СЕМЬ ПРИНЦИПОВ ТЕСТИРОВАНИЯ

- Принцип 1 – Тестирование демонстрирует наличие дефектов
- Принцип 2 – Исчерпывающее тестирование невозможно
- Принцип 3 – Раннее тестирование необходимо
- Принцип 4 – Скопление дефектов

Небольшое количество модулей содержат большинство дефектов, выявленных в ходе тестирования, или демонстрируют наибольшее количество операционных сбоев.

Это еще одно проявление правила Парето 80/20 – 80% дефектов находятся в 20% функций.

СЕМЬ ПРИНЦИПОВ ТЕСТИРОВАНИЯ

- Принцип 1 – Тестирование демонстрирует наличие дефектов
- Принцип 2 – Исчерпывающее тестирование невозможно
- Принцип 3 – Раннее тестирование необходимо
- Принцип 4 – Скопление дефектов
- Принцип 5 – «Парадокс пестицида» (DDT paradox)

Если одни и те же тесты повторяются снова и снова, в конце концов с их помощью вы перестанете находить дефекты.

СЕМЬ ПРИНЦИПОВ ТЕСТИРОВАНИЯ

- Принцип 1 – Тестирование демонстрирует наличие дефектов
- Принцип 2 – Исчерпывающее тестирование невозможно
- Принцип 3 – Раннее тестирование необходимо
- Принцип 4 – Скопление дефектов
- Принцип 5 – «Парадокс пестицида» (DDT paradox)
- Принцип 6 – Тестирование зависит от контекста

Тестирование проводится по-разному в различных контекстах.

Контекст включает: тип продукта, его цели, связанные риски, доступные инструменты, ресурсы и время, опыт команды и т.д.

СЕМЬ ПРИНЦИПОВ ТЕСТИРОВАНИЯ

- Принцип 1 – Тестирование демонстрирует наличие дефектов
- Принцип 2 – Исчерпывающее тестирование невозможно
- Принцип 3 – Раннее тестирование необходимо
- Принцип 4 – Скопление дефектов
- Принцип 5 – «Парадокс пестицида» (DDT paradox)
- Принцип 6 – Тестирование зависит от контекста
- Принцип 7 – Заблуждение об отсутствии ошибок

Нахождение и исправление дефектов не поможет, если разработанная система не удовлетворяет нуждам и ожиданиям пользователей.

Продукт обязан выполнять ПОЛЕЗНУЮ для пользователя задачу.

Тестирование программного обеспечения (software testing) – это

Тестирование программного обеспечения— проверка соответствия между реальным и ожидаемым поведением программы, осуществляемая на конечном наборе тестов, выбранном определенным образом.

QA ≠ QC ≠ Testing

- **Обеспечение качества** (Quality Assurance) – совокупность мероприятий, охватывающих все технологические этапы разработки, выпуска и эксплуатации ПО информационных систем, предпринимаемых на разных стадиях жизненного цикла ПО, для обеспечения качества выпускаемого продукта.

- **Контроль качества** (Quality Control) – совокупность мероприятий проводимых в процессе разработки, для постоянного получения исчерпывающей информации о соответствии объекта тестирования поставленным требованиям.

- **Тестирование ПО** (Testing)– процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определённым образом.

Верификация vs Валидация

- **Верификация** (verification) – это процесс оценки системы или её компонентов с целью определения удовлетворяют ли результаты текущего этапа разработки условиям, которые сформированы в начале этого этапа.

- **Валидация** (validation) – это определение соответствия разрабатываемого ПО ожиданиям и потребностям пользователя, его требованиям к системе.

План тестирования (test plan) – это

Документ, описывающий цели, подходы, ресурсы и график запланированных тестовых активностей и определяющий:

- что тестировать
- что не нужно тестировать
- кто будет тестировать
- где это нужно тестировать и на каком оборудовании
- методы и подходы для проектирования тестов
- критерии для начала и окончания тестирования, а также любые риски, требующие планирования на случай чрезвычайных обстоятельств

Чек-лист (check-list) – высокоуровневый список

- список, который содержит проверки
- список того, что мы собрались сделать
- список того, что хотим не забыть
- список того, что будем проверять
- набор идей тестов

Тест-кейс (test case) – это

Документ в котором есть

входные данные,

условия выполнения и

ожидаемые результаты,

разработанный с целью проверки того или иного свойства или поведения программного средства.

Набор тестов (test suite) – это

Набор тестов (тест-кейсов), собранных в последовательность для достижения некоторой цели.

Хороший тестовый сценарий всегда следует некоторой логике, например: типичному использованию приложения, удобству тестирования, распределению функций по модулям и т.д.

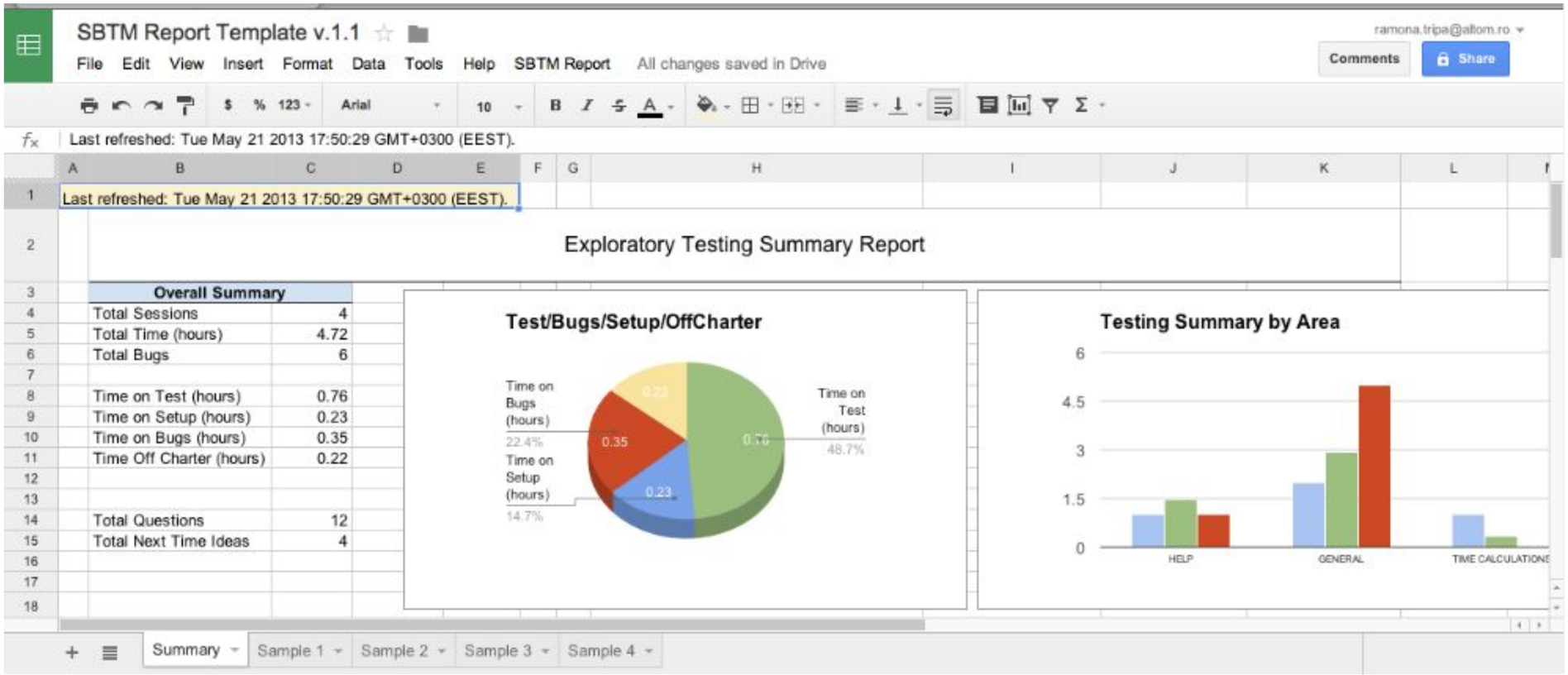
Дефект (defect, bug) – это

Любое несоответствие фактического и ожидаемого результата (согласно требованиям или здравому смыслу).

Изъян в компоненте или системе, который может привести компонент или систему к невозможности выполнить требуемую функцию, например неверный оператор или определение данных.

Отчет о тестировании (test result report, TRR) – это

Документ, подводящий итог проделанной работы в ходе тестирования, а также содержащий оценку состояния качества программы.



Билд («сборка», build) – это

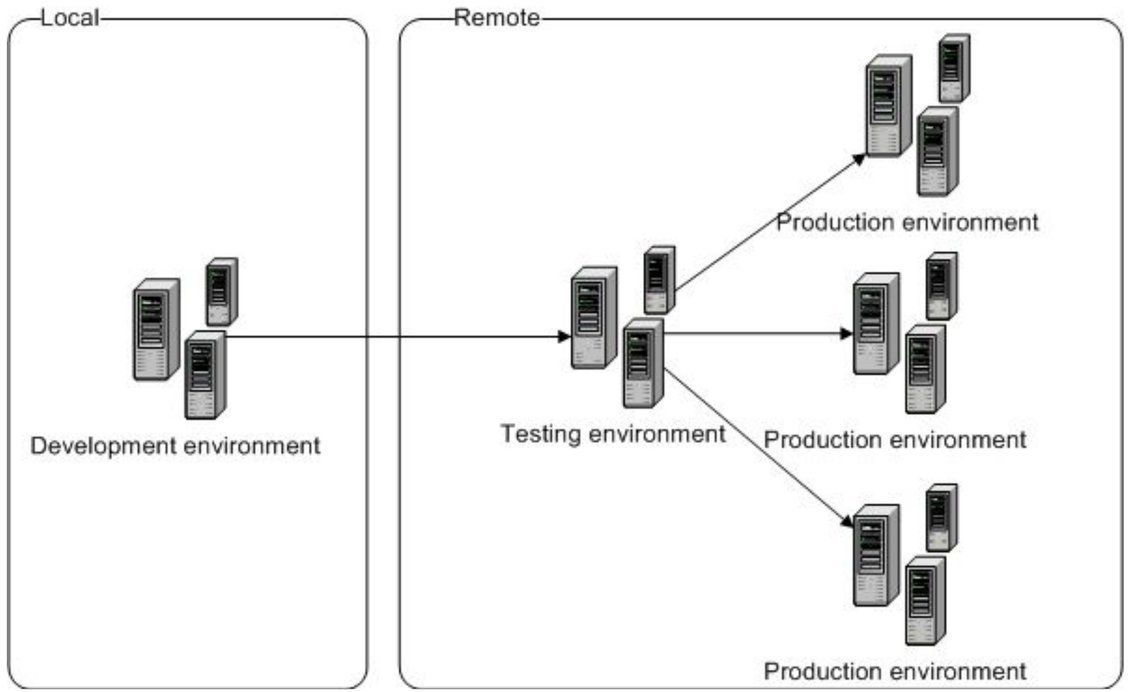
Очередная версия программы.

Финальный билд – часто называют Релизом (Release) – то, что уходит пользователям/заказчику.

Ежедневная сборка (daily build) – действия, в ходе которых система ежедневно (обычно ночью) компилируется и собирается целиком, так что целостная система доступна в любое время, включая все последние изменения.

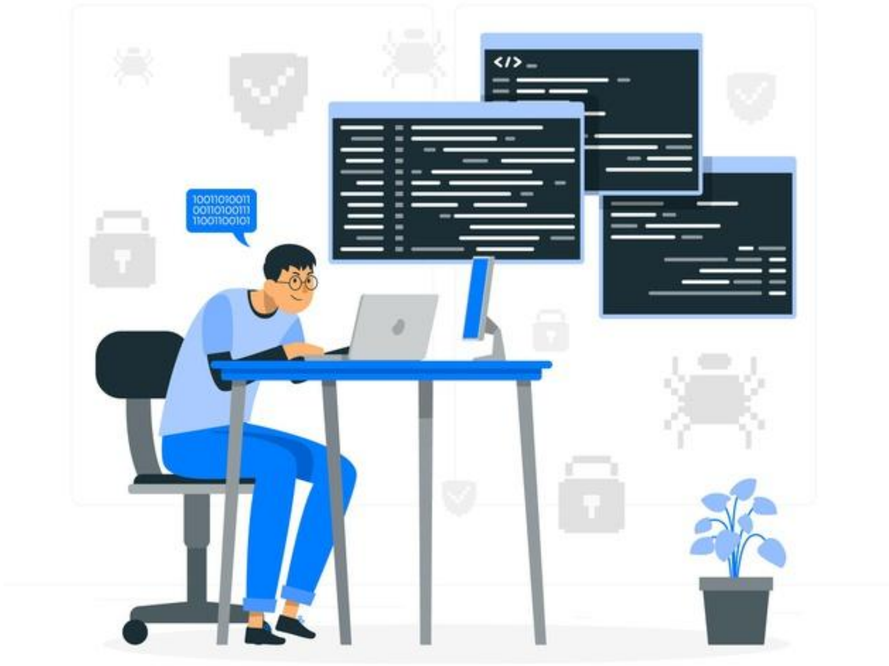
Тестовое окружение (test environment) – это

Аппаратура (по сути компьютер/смартфон и установленное на нем ПО) и инструментарий, необходимые для проведения теста (которыми пользуется тестировщик).



Отладка (Debugging) – это

Процесс поиска, анализа и устранения причин отказов в программном обеспечении.



Качество (Quality) – это

Степень, с которой компонент, система или процесс соответствует зафиксированным требованиям и/или ожиданиям и нуждам пользователя или заказчика.

- если заказчик доволен продуктом – продукт качественный
- если продукт соответствует требованиям – продукт качественный
- у качественного продукта всегда есть преимущества и нет серьёзных недостатков
- хорошее качество – низкий риск потерь (денег, времени, репутации...)
- заказчик должен быть отсатисфачен

Метрика (metric) – это

Шкала измерений и метод, используемый для измерений [ISO 14598].

Варианты метрик:

- покрытие требований тестами – не менее 80%
- плотность покрытия – не менее 3
- закрыто 100% известных критических дефектов,

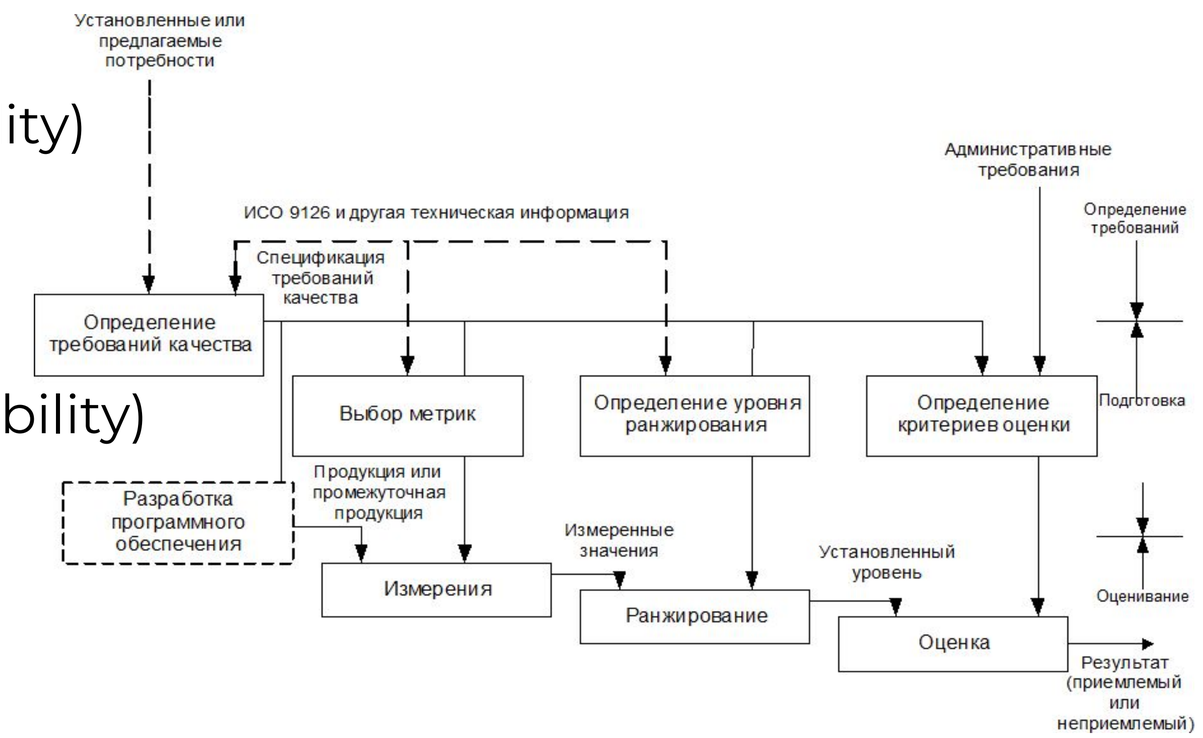
  90% дефектов средней критичности,

  50% остальных дефектов.
- общий показатель прохождения тестов – не менее некоторого значения:

  X = (Passed/Executed)*100%

Качество ПО включает характеристики:

- Функциональная пригодность (Functional suitability)
- Производительность (Performance efficiency)
- Совместимость (Compatibility)
- Удобство использования (Usability)
- Надежность (Reliability)
- Безопасность (Security)
- Ремонтопригодность (Maintainability)
- Переносимость (Portability)



Рекомендуемые ресурсы

- https://www.w3schools.com – множество простой информации по целой серии технологий
- http://www.sql-ex.ru/learn_exercises.php – множество практических заданий по SQL
- https://youtu.be/Z-a7MNStFQs – простой полуторачасовой видеокурс по основам компьютерных сетей
- https://htmlacademy.ru – серия бесплатных курсов по HTML / CSS / JS / PHP
- http://software-testing.ru – большой портал, на котором есть как профессиональные материалы, так и небольшие статьи, понятные и полезные начинающим в помощь