**QA/QC**

**Контроль качества (QC - *quality control*)** — это корректирующий подход, ориентированный на продукт, который сосредотачивается на действиях, поддерживающих достижение надлежащего уровня качества. Тестирование является основным видом контроля качества, в то время как другие виды включают формальные методы (проверку модели и доказательство правильности), моделирование и прототипирование.

**Обеспечение качества (QA - *quality assurance*)** — это превентивный подход, ориентированный на процесс, который сосредотачивается на внедрении и улучшении процессов. Он предполагает, что если правильно следовать хорошему процессу, то будет создан хороший продукт. Обеспечение качества применяется как к процессам разработки, так и к процессам тестирования, и за него несет ответственность каждый участник проекта.

**Тестирование программного обеспечения** — это процесс в рамках жизненного цикла разработки программного обеспечения, который оценивает качество компонента или системы, а также связанных с ними рабочих продуктов. При тестировании эти рабочие продукты называются объектами тестирования. Распространенное заблуждение о тестировании заключается в том, что оно состоит только из выполнения тестов (т.е. запуска программного обеспечения и проверки результатов тестов). Однако тестирование программного обеспечения также включает другие действия и должно быть согласовано с жизненным циклом разработки программного обеспечения

|  |  |
| --- | --- |
| **QA** | **QC** |
| Ориентирован на достижение запрашиваемого качества продукта | Ориентирован на выполнение запрошенного уровня качества ПО |
| Особенная техника управления качеством | Особенная методика проверки качества |
| Не содержит в своей структуре процесса выполнения программы | При любом условии содержит процедуру выполнения программы |
| Всегда ориентирован на процесс | Всегда ориентирован на продукт |
| Базовой целью является своевременное предотвращение багов | Целью является поиск и удаление багов |
| Ответственный за полный жизненный цикл создания ПО | Ответственный за жизненный цикл проверки ПО |
| Ради удовлетворения требования клиента может определять стандарты и методологии | Верифицирует, что при создании ПО соблюдаются все известные стандарты |
| Задействованы все участники команды тестирования и разработки | Участвуют все участники команды тестирования |
| Проводится до начала QC | Выполняется исключительно после завершения QA |

Результаты тестов используются как QA, так и QC. В QC они используются для исправления дефектов, а в QA они обеспечивают обратную связь, насколько хорошо выполняются процессы разработки и тестирования.

Разница между QA-специалистом и тестировщиком в том, что тестировщик только тестирует ту или иную фичу/проблему и проверяет только ее выполнение (или невыполнение), как QA-специалист улучшает продукт, описывает документацию, актуализирует информацию на dev-стендах, составляет отчеты, метрики, прохождение тест-планов.

**QC** — гарантирует, что процессы, установленные QA, соблюдаются. То есть QC - это часть QA - процесс установления стандартов и проверки, что ПО сделано правильно. Цель контроля качества - проверить, соблюдалась ли предписанная модель. Это может быть достигнуто путем проведения аудитов и определения того, следовала ли команда определенной модели для достижения качества.

**Тестирование** — это не только техническая активность. Необходимо также надлежащим образом планировать, управлять, оценивать, отслеживать и контролировать тестирование . Тестировщики используют инструменты , но важно помнить, что тестирование — это в значительной степени интеллектуальная деятельность, требующая от тестировщиков специальных знаний, использования аналитических навыков и применения критического и системного мышления.

**Принципы тестирования**

1. **Тестирование демонстрирует наличие дефектов, а не их отсутствие**. Тестирование может показать наличие дефектов в объекте тестирования, но не может доказать их отсутствие. Тестирование снижает вероятность того, что дефекты в объекте тестирования останутся необнаруженными, но даже если дефекты не были обнаружены, тестирование не доказывает корректности объекта тестирования.

2. **Исчерпывающее тестирование невозможно**. Полное тестирование с использованием всех комбинаций вводов и предусловий физически невыполнимо, за исключением тривиальных случаев. Вместо того, чтобы пытаться провести исчерпывающее тестирование, следует использовать методы тестирования, расстановку приоритетов тестовых сценариев и тестирование, основанное на рисках, чтобы сосредоточить усилия по тестированию.

3. **Раннее тестирование экономит время и деньги**. Дефекты, устраненные на ранней стадии процесса, не вызовут последующих дефектов в производных рабочих продуктах. Стоимость качества будет снижена, так как позже в ЖЦ ПО будет происходить меньше отказов. Для раннего обнаружения дефектов как можно раньше следует начинать как статическое тестирование, так и динамическое тестирование.

4. **Кластеризация дефектов**. Обычно небольшое количество системных компонентов содержит большинство обнаруженных дефектов или порождает большинство эксплуатационных отказов. Это явление является иллюстрацией принципа Парето. Предсказанные и фактические кластеры дефектов, наблюдаемые в ходе тестирования или эксплуатации, являются важными входными данными для тестирования, основанного на рисках.

5. **Тесты устаревают**. Если одни и те же тесты повторяются много раз, они становятся все более неэффективными в обнаружении новых дефектов. Для обнаружения новых дефектов может потребоваться изменение существующих тестов и тестовых данных, а также написание новых тестов. Однако в некоторых случаях повторение одних и тех же тестов может иметь положительный результат, например, при автоматизированном регрессионном тестировании.

6. **Тестирование зависит от контекста**. Не существует единого универсального подхода к тестированию. Тестирование выполняется по-разному в зависимости от контекста.

**7. Заблуждение об отсутствии дефектов**. Было бы ошибкой ожидать, что верификация программного обеспечения обеспечит успех системы. Тщательное тестирование всех указанных требований и исправление всех обнаруженных дефектов может привести к созданию системы, которая не будет соответствовать потребностям и ожиданиям пользователей, не будет помогать в достижении бизнес-целей заказчика и будет уступать другим конкурирующим системам. Также в дополнение к верификации следует проводить валидацию.

**На каком этапе нужно вводить QA/QC**

Тестировщик вступает уже после этапа разработки

Статическое тестирование позволяет оценить качество продукта и приобрести в нем уверенность. Верификация задокументированных требований позволяет заинтересованным сторонам убедиться, что они отражают их реальные требования. Поскольку статическое тестирование может быть выполнено на ранних этапах жизненного цикла разработки программного обеспечения, можно создать общее понимание среди заинтересованных сторон, участвующих в этом процессе. Уровень взаимодействия также улучшится. По этой причине рекомендуется привлекать все заинтересованные стороны на этапе статического тестирования (ISTQB 3.1.2 «Ценность статического тестирования»)

Активности QA проходят на всем протяжении жизненного цикла ПО: на этапе построения, анализа и улучшения процессов, формирования релизных политик, риск менеджмента и т.д.

QC подключаются на этапе составления критериев качества, quality gate-ов, метрик и способов оценки.

**Карьерный рост**

С позиции QA-специалиста можно вырасти:

**QA Lead (**Менеджер по тестированию) **–** это специалист, который руководит командой (командами) тестировщиков

**QA Auto** **–** это специалист, который пишет тесты на основе скриптов для автоматизации тестирования

**QA Fullstack –** это квалифицированный специалист, который объединяет в себе знания, умения и навыки QA инженера и автоматизатора тестирования

**QA Mobile** **–** это специалист, который занимается тестированием мобильных приложений

**QA Load & Performance** **–** тип тестирования, в котором мы проверяем, соответствует ли наша система поставленным нефункциональным требованиям при работе под высокой нагрузкой

**Backend разработчик –** это программист, который создает и поддерживает нужные механизмы, они, в свою очередь, обрабатывают данные и выполняют определенные действия на сайтах

**Frontend разработчик** - это специалист, который создает интерфейсы веб-сайтов и приложений

**Технический писатель** - это специалист, который создает документацию по сложному, технологическому продукту

**Менеджер по продукту** — это человек, определяющий потребности клиентов и более крупные бизнес-цели, которые можно будет достичь с помощью продукта или возможности

**Литература**

* [**https://www.youtube.com/@rusau**](https://www.youtube.com/@rusau) **– Ютуб канал о тестировании**
* [**https://obuchalka.org/20201103126550/testirovanie-dot-kom-ili-posobie-po-jestokomu-obrascheniu-s-bagami-v-internet-startapah-savin-r-2007.html**](https://obuchalka.org/20201103126550/testirovanie-dot-kom-ili-posobie-po-jestokomu-obrascheniu-s-bagami-v-internet-startapah-savin-r-2007.html) **- Тестирование Дот Ком, или Пособие по жестокому обращению с багами в интернет-стартапах, Савин Р., 2007**
* [**https://stepik.org/course/171826/promo#toc**](https://stepik.org/course/171826/promo#toc) **– Курс на stepik**
* [**https://vladislaveremeev.gitbook.io/qa\_bible**](https://vladislaveremeev.gitbook.io/qa_bible) **– QA Bible**
* [**https://roadmap.sh/qa**](https://roadmap.sh/qa) **- RoadMap QA**
* [**http://thetestingmap.org/**](http://thetestingmap.org/) **- Testing Map**
* [**http://wiki.software-testing.ru/**](http://wiki.software-testing.ru/%D0%97%D0%B0%D0%B3%D0%BB%D0%B0%D0%B2%D0%BD%D0%B0%D1%8F_%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0) **- Русская вики по тестированию**
* [**https://software-testing.ru/library/testing/testing-tools**](https://software-testing.ru/library/testing/testing-tools) **- software-testing**
* [**https://www.protesting.ru/**](https://www.protesting.ru/) **- ПроТестинг**
* [**https://habr.com/ru/companies/yandex\_praktikum/articles/760304/**](https://habr.com/ru/companies/yandex_praktikum/articles/760304/) **- Статья про тестирование**