Лекция 4: Тестирование и тестировщики

1.1. Что такое тестирование и откуда оно появилось

В первую очередь дадим определение тестирования ПО, чтобы чётче понимать, о чём пойдёт речь.

Тестирование программного обеспечения — процесс анализа программного средства и сопутствующей документации с целью выявления дефектов и повышения качества продукта.

В глоссарии ISTQB[1] нет термина «тестирование ПО», который широко используется в русском языке. Там есть лишь термин «тестирование (testing[2])».

На протяжении десятилетий развития разработки ПО к вопросам тестирования и обеспечения качества подходили очень и очень по-разному. Можно выделить несколько основных «эпох тестирования».

В 50–60-х годах прошлого века процесс тестирования был предельно формализован, отделён от процесса непосредственной разработки ПО и «математизирован». Фактически тестирование представляло собой скорее отладку программ (debugging[3]). Существовала концепция т.н. «исчерпывающего тестирования (exhaustive testing[4])» — проверки всех возможных путей выполнения кода со всеми возможными входными данными. Но очень скоро было выяснено, что исчерпывающее тестирование невозможно, т.к. количество возможных путей и входных данных очень велико, а также при таком подходе сложно найти проблемы в документации.

Задание 1.1.a: представьте, что ваша программа по трём введённым целым числам определяет, может ли существовать треугольник с такими длинами сторон. Допустим, что ваша программа выполняется в некоей изолированной идеальной среде, и вам всего-то осталось проверить корректность её работы на трёх 8-байтовых знаковых целых числах. Вы используете автоматизацию, и компьютер может провести 100 миллионов проверок в секунду. Сколько займёт проверка всех вариантов?

А задумались ли вы, как подготовить для этого теста проверочные данные (на основе которых можно определить, верно ли сработала программа в каждом конкретном случае)?

В 70-х годах фактически родились две фундаментальные идеи тестирования: тестирование сначала рассматривалось как процесс доказательства работоспособности программы в некоторых заданных условиях (positive testing[5]), а затем — строго наоборот: как процесс доказательства неработоспособности программы в некоторых заданных условиях (negative testing[6]). Это внутреннее противоречие не только не исчезло со временем, но и в наши дни многими авторами совершенно справедливо отмечается как две взаимодополняющие цели тестирования.

---

[1] International Software Testing Qualifications Board Glossary. [http://www.istqb.org/downloads/glossary.html]

[2] **Testing.** The process consisting of all lifecycle activities, both static and dynamic, concerned with planning, preparation and evaluation of software products and related work products to determine that they satisfy specified requirements, to demonstrate that they are fit for purpose and to detect defects. [ISTQB Glossary]

[3] **Debugging.** The process of finding, analyzing and removing the causes of failures in software. [ISTQB Glossary]

[4] **Complete testing, exhaustive testing.** A test approach in which the test suite comprises all combinations of input values and preconditions. [ISTQB Glossary]

[5] **Positive Testing.** Testing aimed at showing software works. Also known as «test to pass». [aptest.com]

[6] **Negative testing.** Testing aimed at showing software does not work. Also known as «test to fail». [aptest.com]

Отметим, что «*процесс доказательства неработоспособности программы*» ценится чуть больше, т.к. не позволяет закрывать глаза на обнаруженные проблемы.

Внимание! Скорее всего, именно из этих рассуждений проистекает **неверное** понимание того, что негативные тест-кейсы должны заканчиваться возникновением сбоев и отказов в приложении. Нет, это не так. Негативные тест-кейсы пытаются вызвать сбои и отказы, но корректно работающее приложение выдерживает это испытание и продолжает работать верно. Также отметим, что ожидаемым результатом негативных тест-кейсов является именно корректное поведение приложения, а сами негативные тест-кейсы считаются пройденными успешно, если им не удалось «поломать» приложение. (См. подробности в главе «Чек-листы, тест-кейсы, наборы тест-кейсов»{110}).

Много «классики тестирования» можно почерпнуть из книги «Искусство тестирования программ» Гленфорда Майерса (издания 1979, 2004, 2011 годов). Однако большинство критиков отмечает, что эта книга мало подходит для начинающих и куда больше ориентирована на программистов, чем на тестировщиков. Что, впрочем, не умаляет её ценности. В оригинале книга называется «The art of software testing» (Glenford J. Myers).

Итак, ещё раз самое важное, что тестирование «приобрело» в 70-е годы:

- тестирование позволяет удостовериться, что программа соответствует требованиям;
- тестирование позволяет определить условия, при которых программа ведёт себя некорректно.

В 80-х годах произошло ключевое изменение места тестирования в разработке ПО: вместо одной из финальных стадий создания проекта тестирование стало применяться на протяжении всего цикла разработки (software lifecycle[7]) (также см. описание итерационной инкрементальной модели разработки ПО в главе «Модели разработки ПО»{18}), что позволило в очень многих случаях не только быстро обнаруживать и устранять проблемы, но даже предсказывать и предотвращать их появление.

В этот же период времени отмечено бурное развитие и формализация методологий тестирования и появление первых элементарных попыток автоматизировать тестирование.

В 90-х годах произошёл переход от тестирования как такового к более всеобъемлющему процессу, который называется «обеспечение качества (quality assurance[8])», охватывает весь цикл разработки ПО и затрагивает процессы планирования, проектирования, создания и выполнения тест-кейсов, поддержку имеющихся тест-кейсов и тестовых окружений. Тестирование вышло на качественно новый уровень, который естественным образом привёл к дальнейшему развитию методологий, появлению достаточно мощных инструментов управления процессом тестирования и инструментальных средств автоматизации тестирования, уже вполне похожих на своих нынешних потомков.

[7] **Software lifecycle.** The period of time that begins when a software product is conceived and ends when the software is no longer available for use. The software lifecycle typically includes a concept phase, requirements phase, design phase, implementation phase, test phase, installation and checkout phase, operation and maintenance phase, and sometimes, retirement phase. Note these phases may overlap or be performed iteratively. [ISTQB Glossary]

[8] **Quality assurance.** Part of quality management focused on providing confidence that quality requirements will be fulfilled. [ISTQB Glossary]

Хорошим источником дополнительной информации о процессах тестирования является книга Рекса Блэка «Ключевые процессы тестирования» («Critical Testing Processes», Rex Black).

В нулевые годы нынешнего века развитие тестирования продолжалось в контексте поиска всё новых и новых путей, методологий, техник и подходов к обеспечению качества. Серьёзное влияние на понимание тестирования оказало появление гибких методологий разработки и таких подходов, как «разработка под управлением тестированием[9] (test-driven development[10], TDD)». Автоматизация тестирования уже воспринималась как обычная неотъемлемая часть большинства проектов, а также стали популярны идеи о том, что во главу процесса тестирования следует ставить не соответствие программы требованиям, а её способность предоставить конечному пользователю возможность эффективно решать свои задачи.

О **современном этапе** развития тестирования мы будем говорить на протяжении всего остального материала. Если же отметить вкратце основные характеристики, то получится примерно такой список: гибкие методологии и гибкое тестирование, глубокая интеграция с процессом разработки, широкое использование автоматизации, колоссальный набор технологий и инструментальных средств, кросс-функциональность команды (когда тестировщик и программист во многом могут выполнять работу друг друга).

Воистину подробнейшую историю развития тестирования ПО (начиная с 1822 года, не шутка) можно найти в статье «The History of Software Testing»[11] на ресурсе «Testing References». Также немалый интерес представляет статья «The Growth of Software Testing»[12] (David Gelperin, Bill Hetzel).

Задание 1.1.b: если вам не очень хорошо знакомы такие понятия как TDD, BDD, DDT, KDT, — найдите их описание в Интернете и изучите. Конечно же, это задание относится и к любым другим непонятным терминам.

[9] Да, грамматически корректно будет «разработка под управлением тестирования», но традиционно так сложилось, что целый ряд подобных подходов «под управлением…» называют, используя творительный падеж: т.е. «…тестированием», «…данными», «…ключевыми словами» и т.д.

[10] **Test-driven development.** A way of developing software where the test cases are developed, and often automated, before the software is developed to run those test cases. [ISTQB Glossary]

[11] «The History of Software Testing» [http://www.testingreferences.com/testinghistory.php]

[12] «The Growth of Software Testing», David Gelperin, Bill Hetzel [http://www.clearspecs.com/downloads/ClearSpecs16V01_GrowthOfSoftwareTest.pdf]

1.2. Кто такой тестировщик и что он делает

Если поискать информацию по ключевым фразам из названия этой главы, можно найти уйму совершенно противоречивых ответов. И дело здесь в первую очередь в том, что авторы большинства «должностных обязанностей» приписывают всей профессии некий утрированный набор характеристик отдельных её представителей.

В то же время даже в ЕКСД разделены должности «специалиста по тестированию программного обеспечения» и «тестировщика программного обеспечения».

Если вам интересно, почитайте документ «Постановление Министерства труда и социальной защиты Республики Беларусь № 148 от 15 декабря 2009 г. О внесении изменений и дополнений в выпуск 1 Единого квалификационного справочника должностей служащих (ЕКСД)».

Теперь вернёмся к исходному вопросу и посмотрим на него с двух точек зрения: какова квалификация тестировщика, и где он работает.

Упрощённо отразим это в таблице 1.2.a.

Таблица 1.2.a — Типичные виды деятельности тестировщика.

	Небольшие фирмы	**Большие фирмы**
Низкая квалификация	Подмастерье, часто предоставленный сам себе в решении задач.	Рядовой участник проектов, одновременно проходящий интенсивное повышение квалификации.
Высокая квалификация	Мастер на все руки с богатым, но не всегда структурированным опытом.	Эксперт в одной или нескольких областях, консультант, руководитель направления.

Поскольку чем выше квалификация специалиста{275}, тем шире его выбор мест работы (даже в рамках одной крупной фирмы), основное внимание уделим именно квалификационным особенностям работы тестировщика.

В начале карьеры любой специалист (и тестировщик не является исключением) является исполнителем и учеником. Достаточно хорошо понимать, что такое тест-кейсы, отчёты о дефектах, уметь читать требования, пользоваться парой инструментальных средств и хорошо уживаться в команде.

Постепенно тестировщик начинает погружаться во все стадии разработки проекта, понимая их всё полнее и полнее, начинает не только активно использовать, но и разрабатывать проектную документацию, принимать всё более ответственные решения.

Если выразить образно главную цель тестировщика, то она будет звучать так: «понимать, что в настоящий момент необходимо проекту, получает ли проект это необходимое в должной мере, и если нет, как изменить ситуацию к лучшему». Звучит похоже на цель руководителя проекта, верно? Верно. Начиная с некоторого уровня развития, IT-специалисты, по большому счёту, различаются лишь наборами технических навыков и основной областью приложения этих навыков.

Так какие же технические навыки нужны, чтобы успешно начать работать тестировщиком? Прежде чем приступить к самому перечислению, оговорим особо: этот список рассчитан в первую очередь на тех, кто приходит в тестирование из нетехнических профессий (хотя часто его же приходится озвучивать и студентам технических вузов).

0) Знание иностранных языков. Да, это нетехнический навык. Но тем не менее он идёт под номером «ноль». Можете считать это аксиомой: «нет знания английского — нет карьеры в IT». Другие иностранные языки тоже приветствуются, но английский первичен.

Задание 1.2.a: если вы сомневаетесь в том, достаточен ли ваш уровень английского, проверьте себя: если вы можете без труда читать технические статьи хотя бы в Википедии, минимально достаточный уровень у вас есть.

1) Уверенное владение компьютером на уровне по-настоящему продвинутого пользователя и желание постоянно развиваться в этой области. Можете ли вы представить себе профессионального повара, который не может пожарить картошку (не «не обязан», а «не умеет в принципе»)? Выглядит странно? Не менее странно выглядит «IT'шник» (именно так, в кавычках), неспособный набрать вменяемо отформатированный текст, скопировать файл по сети, развернуть виртуальную машину или выполнить любое иное повседневное рутинное действие.
2) Программирование. Оно на порядки упрощает жизнь любому IT'шнику — и тестировщику в первую очередь. Можно ли тестировать без знания программирования? Да, можно. Можно ли это делать по-настоящему хорошо? Нет. И сейчас самый главный (почти религиозно-философский) вопрос: какой язык программирования изучать? C/C++/C#, Java, PHP, JavaScript, Python, Ruby и т.д. — начинайте с того, на чём написан ваш проект. Если проекта пока ещё нет, начинайте с JavaScript (на текущий момент — самое универсальное решение).
3) Базы данных и язык SQL. Здесь от тестировщика тоже не требуется квалификация на уровне узких специалистов, но минимальные навыки работы с наиболее распространёнными СУБД и умение писать простые запросы можно считать обязательными.
4) Понимание принципов работы сетей и операционных систем. Хотя бы на минимальном уровне, позволяющем провести диагностику проблемы и решить её своими силами, если это возможно.
5) Понимание принципов работы веб-приложений и мобильных приложений. В наши дни почти всё пишется именно в виде таких приложений, и понимание соответствующих технологий становится обязательным для эффективного тестирования.

Надеюсь, вы обратили внимание на то, что самого тестирования в списке нет. Всё верно, ведь ему посвящена вся эта книга целиком, так что позволим себе не копировать её сюда.

В завершение главы также отметим личностные качества, позволяющие тестировщику быстрее стать отличным специалистом:

1) повышенная ответственность и исполнительность;
2) хорошие коммуникативные навыки, способность ясно, быстро, чётко выражать свои мысли;
3) терпение, усидчивость, внимательность к деталям, наблюдательность;
4) хорошее абстрактное и аналитическое мышление;
5) способность ставить нестандартные эксперименты, склонность к исследовательской деятельности.

Да, сложно найти человека, который бы в равной мере обладал всеми перечисленными качествами, но всегда полезно иметь некий ориентир для саморазвития.

Очень часто можно услышать вопрос о том, обязательно ли тестировщику иметь техническое высшее образование. Не обязательно. Хотя при его наличии на первых этапах карьеры, конечно, легче. Но со временем разница между теми, у кого такое образование есть, и теми, у кого нет, становится практически незаметной.

1.3. Что нужно знать и уметь и чему можно научиться

В предыдущей главе мы осознанно не обсуждали конкретный перечень необходимых начинающему тестировщику знаний и умений, т.к. он заслуживает отдельного рассмотрения.

Показанные ниже данные представляют собой адаптированную выжимку из карты компетенций тестировщика. Все навыки здесь условно разделены на три группы:

- Профессиональные — это именно «тестировщицкие», ключевые навыки, отличающие тестировщика от других IT-специалистов.
- Технические — это общие навыки в области IT, которыми тем не менее должен обладать и тестировщик.
- Личностные — в русском языке термин «soft skills» часто переводят как «навыки межличностного общения», но исходный термин несколько шире.

Задание 1.3.a: в процессе чтения приведённых здесь перечней компетенций отмечайте непонятные вещи, ищите дополнительную информацию и добивайтесь у себя понимания описанного хотя бы на уровне «знаю, о чём идёт речь».

Профессиональные навыки

Таблица 1.3.a — Профессиональные навыки тестировщика

Предметная область	Начальный уровень	Уровень младшего или среднего специалиста
Процессы тестирования и разработки программного обеспечения		
Процесс тестирования ПО	Этому вопросу посвящена глава «Процессы тестирования и разработки ПО»{18}	Глубокое понимание стадий процесса тестирования, их взаимосвязи и взаимовлияния, умение планировать собственную работу в рамках полученного задания в зависимости от стадии тестирования
Процесс разработки ПО		Общее понимание моделей разработки ПО, их связи с тестированием, умение расставлять приоритеты в собственной работе в зависимости от стадии развития проекта
Работа с документацией		
Анализ требований	Этому вопросу посвящена глава «Тестирование документации и требований»{29}	Умение определять взаимосвязи и взаимозависимость между различными уровнями и формами представления требований, умение формулировать вопросы с целью уточнения неясных моментов
Тестирование требований		Знание свойств хороших требований и наборов требований, умение анализировать требования с целью выявления их недостатков, умение устранять недостатки в требованиях, умение применять техники повышения качества требований
Управление требованиями	Не требуется	Общее понимание процессов выявления, документирования, анализа и модификации требований
Бизнес-анализ		Общее понимание процессов выявления и документирования различных уровней и форм представления требований

Предметная область	Начальный уровень	Уровень младшего или среднего специалиста
Оценка и планирование		
Создание плана тестирования	Эти вопросы частично затронуты в главе «Оценка трудозатрат, планирование и отчётность»{203}, но их глубокое понимание требует отдельного длительного изучения	Общее понимание принципов планирования в контексте тестирования, умение использовать готовый тест-план для планирования собственной работы
Создание стратегии тестирования		Общее понимание принципов построения стратегии тестирования, умение использовать готовую стратегию для планирования собственной работы
Оценка трудозатрат		Общее понимание принципов оценки трудозатрат, умение оценивать собственные трудозатраты при планировании собственной работы
Работа с тест-кейсами		
Создание чек-листов	Этому вопросу посвящена глава «Чек-листы, тест-кейсы, наборы тест-кейсов»{110}	Твёрдое умение использовать техники и подходы к проектированию тестовых испытаний, умение декомпозировать тестируемые объекты и поставленные задачи, умение создавать чек-листы
Создание тест-кейсов		Твёрдое умение оформлять тест-кейсы согласно принятым шаблонам, умение анализировать готовые тест-кейсы, обнаруживать и устранять имеющиеся в них недостатки
Управление тест-кейсами	Не требуется	Общее понимание процессов создания, модификации и повышения качества тест-кейсов
Методологии тестирования		
Функциональное и доменное тестирование	Этому вопросу посвящена глава «Подробная классификация тестирования»{64}	Знание видов тестирования, твёрдое умение использовать техники и подходы к проектированию тестовых испытаний, умение создавать чек-листы и тест-кейсы, умение создавать отчёты о дефектах
Тестирование интерфейса пользователя	Не требуется	Умение проводить тестирование интерфейса пользователя на основе готовых тестовых сценариев или в рамках исследовательского тестирования
Исследовательское тестирование		Общее умение использовать матрицы для быстрого определения сценариев тестирования, общее умение проводить новые тесты на основе результатов только что выполненных
Интеграционное тестирование		Умение проводить интеграционное тестирование на основе готовых тестовых сценариев
Локализационное тестирование		Умение проводить локализационное тестирование на основе готовых тестовых сценариев
Инсталляционное тестирование		Умение проводить инсталляционное тестирование на основе готовых тестовых сценариев
Регрессионное тестирование		Общее понимание принципов организации регрессионного тестирования, умение проводить регрессионное тестирование по готовым планам
Работа с отчётами о дефектах		
Создание отчётов о дефектах	Этому вопросу посвящена глава «Отчёты о дефектах»{162}	Твёрдое знание жизненного цикла отчёта об ошибке, твёрдое умение создавать отчёты о дефектах согласно принятым шаблонам, умение анализировать готовые отчёты, обнаруживать и устранять имеющиеся в них недостатки
Анализ причин возникновения ошибки	Не требуется	Базовое умение исследовать приложение с целью выявления источника (причины) ошибки, элементарное умение формировать рекомендации по устранению ошибки
Использование баг-трекинговых систем		Умение использовать баг-трекинговые системы на всех стадиях жизненного цикла отчётов о дефектах

Предметная область	Начальный уровень	Уровень младшего или среднего специалиста
Работа с отчётами о результатах тестирования		
Создание отчётов о результатах тестирования	Не требуется, но частично рассмотрено в главе «Оценка трудозатрат, планирование и отчётность»{203}	Умение предоставлять необходимую информацию для формирования отчёта о результатах тестирования, умение анализировать готовые отчёты о результатах тестирования с целью уточнения планирования собственной работы

Технические навыки

Таблица 1.3.b — Технические навыки тестировщика.

Предметная область	Начальный уровень	Уровень младшего или среднего специалиста
Операционные системы		
Windows	Использование на уровне уверенного пользователя	Установка, использование и администрирование, решение проблем, конфигурирование с целью настройки тестового окружения и выполнения тест-кейсов
Linux	Общее знакомство	Установка, использование и администрирование, решение проблем, конфигурирование с целью настройки тестового окружения и выполнения тест-кейсов
Mac OS	Не требуется	Общее знакомство
Виртуальные машины	Использование на уровне начинающего пользователя	Установка, использование и администрирование, решение проблем, конфигурирование с целью настройки тестового окружения и выполнения тест-кейсов
Базы данных		
Реляционная теория	Не требуется	Общее понимание, умение читать и понимать схемы баз данных в общепринятых графических нотациях
Реляционные СУБД		Умение устанавливать, настраивать и использовать для настройки тестового окружения и выполнения тест-кейсов
Язык SQL		Умение писать и выполнять простые запросы с использованием инструментальных средств работы с БД/СУБД[13]
Компьютерные сети		
Сетевые протоколы	Не требуется	Общее понимание принципов работы стека TCP/IP, умение конфигурировать локальные сетевые настройки операционной системы
Сетевые утилиты		Общее понимание и умение использовать утилиты диагностики состояния и неполадок в сети
Веб-технологии		
Веб-серверы	Не требуется	Общее понимание принципов работы веб-серверов, умение устанавливать и настраивать
Серверы приложений		Общее понимание принципов работы серверов приложений, умение устанавливать и настраивать
Веб-сервисы		Общее понимание принципов работы веб-сервисов и способов диагностики неполадок в их работе
Языки разметки	Общее представление об HTML и CSS	Умение использовать HTML и CSS для создания простых страниц

[13] «Работа с MySQL, MS SQL Server и Oracle в примерах», Святослав Куликов [http://svyatoslav.biz/database_book/]

Предметная область	Начальный уровень	Уровень младшего или среднего специалиста
Протоколы передачи данных	Не требуется	Общее понимание принципов работы протоколов прикладного уровня OSI-модели, общее понимание принципов диагностики возникших неполадок
Языки веб-программирования		Начальные знания хотя бы в одном языке программирования, используемом для создания веб-приложений
Мобильные платформы и технологии		
Android	Не требуется	Использование на уровне начинающего пользователя
iOS		Использование на уровне начинающего пользователя
Windows Phone		Использование на уровне начинающего пользователя

Личностные навыки

Таблица 1.3.c — Личностные навыки тестировщика.

Предметная область	Начальный уровень	Уровень младшего или среднего специалиста
Коммуникативные навыки		
Деловое использование e-mail	Минимальные навыки	Понимание и строгое следование правилам делового общения с использованием e-mail и сервисов мгновенных сообщений
Устное деловое общение	Минимальные навыки	Понимание и строгое следование правилам устного делового общения
Прохождение собеседований	Не требуется	Начальный опыт прохождения собеседований
Навыки самоорганизации		
Планирование собственного времени	Минимальные навыки, общие представления	Развитые навыки планирования собственного времени, умение пользоваться соответствующими инструментами, умение оценивать трудозатраты в рамках полученных заданий
Отчётность о своей работе	Начальные навыки	Развитые навыки отчётности о своей работе, умение пользоваться соответствующими инструментами

Возможно, вы заметили, что в этом перечне навыков нет отдельного списка, посвящённого автоматизации тестирования. Он не включён в данную книгу по трём причинам: а) он огромен; б) он постоянно меняется; в) эта книга всё же о тестировании вообще, хоть в ней и есть краткие сведения об автоматизации (см. раздел «Автоматизация тестирования»{252}). Если же сказать в двух словах, то автоматизатор должен знать всё то же, что и «классический» тестировщик, а также уметь программировать на 3–5 языках — хотя бы немного. И всё. Инструменты на начальном уровне можно освоить за несколько дней.

1.4. Мифы и заблуждения о тестировании

Возможно, здесь вы ожидали прочитать нечто наподобие «Семи бед тестирования» Джеймса Виттакера (см. ниже). Нет, здесь будут «мифы», которые актуальны не для состоявшихся профессионалов, а для новичков и тех, кто ещё только собирается обучаться тестированию.

Текст этой главы составлен в основном по итогам бесед со слушателями тренингов, а если точнее — по фразам, начинающимся с «а я думал(а), что…» или «а правда ли, что…»

Обязательно почитайте прекрасный цикл статей «The 7 Plagues of Software Testing»[14] (James Whittaker).

Итак: «А я думал(а), что…» / «А правда ли, что…»

Не надо разбираться в компьютерах

Без комментариев. Нет, возможно, существуют некие ничтожно малые доли процента деятельности тестировщика, которую можно реализовать «на пальцах». Но этой бесконечно малой величиной можно пренебречь.

Обязательно надо хорошо знать программирование

Очень больно относить эту мысль к мифам. Хорошо, когда тестировщик знает программирование. Ещё лучше, когда он знает его хорошо. Но даже общих отдалённых представлений о программировании хватает для начала карьеры. А дальше уже — по обстоятельствам.

В тестировании всё просто

Если развить аналогию, то и в кулинарии всё просто, если мы говорим о заваривании чая в пакетике. Но как подобным чаем не заканчивается кулинария, так и тестирование не заканчивается случаями «ой, тут вот картинка не загрузилась». Даже на исключительно практическом уровне задачи тестирования могут быть сопоставимы по сложности с задачами проектирования и разработки программ (хм, почему ж нет мифа «программирование — это просто», ведь «Hello, world» написать не тяжело). А если мы посмотрим на «надёжность программного обеспечения» с научной точки зрения, то перспективы роста сложности вообще ничем не ограничены. Обязательно ли каждому тестировщику «лезть в эти дебри»? Нет. Но если хочется — можно. К тому же это очень интересно.

В тестировании куча рутины и скуки

Не больше и не меньше, чем в иных IT-профессиях. Остальное зависит от конкретного тестировщика и того, как он организует свою работу.

[14] «The plague of repetitiveness», James Whittaker [http://googletesting.blogspot.com/2009/06/by-james.html]
«The Plague of Amnesia», James Whittaker [http://googletesting.blogspot.com/2009/07/plague-of-amnesia.html]
«The Plague of Boredom», James Whittaker [http://googletesting.blogspot.com/2009/07/plague-of-boredom.html]
«The Plague of Homelessness», James Whittaker [http://googletesting.blogspot.com/2009/07/plague-of-homelessness.html]
«The Plague of Blindness», James Whittaker [http://googletesting.blogspot.com/2009/07/plague-of-blindness.html]
«The 7th Plague and Beyond», James Whittaker [http://googletesting.blogspot.com/2009/09/7th-plague-and-beyond.html]
«The Plague of Entropy», James Whittaker [http://googletesting.blogspot.com/2009/09/plague-of-entropy.html]

Тестировщика должны всему-всему научить

Не должны. И уж тем более «всему-всему». Да, если мы говорим о явно обозначенном учебном процессе, то его организаторы (будь то предмет в университете, учебный курс в некоем тренинговом центре или отдельный тренинг внутри компании) часто берут на себя определённые «педагогические обязательства». Но подобная учебная деятельность никогда не заменит саморазвития (хотя и может в нужный момент помочь в выборе направления пути). IT-отрасль меняется очень интенсивно и непрерывно. Учить**ся** придётся до пенсии.

В тестировщики идут те, кто не смог стать программистом

А в скрипачи — те, кто не смог стать пианистом? Я думаю, что некий небольшой процент «не ставших программистами» в тестировании есть. Но он теряется на фоне тех, кто шёл в тестирование изначально и сознательно, а также тех, кто пришёл в тестирование из программирования.

В тестировании сложно построить карьеру

При должном старании карьера в тестировании оказывается едва ли не самой динамичной (по сравнению с другими IT-направлениями). Тестирование само по себе — очень бурно развивающаяся отрасль IT, и здесь всегда можно выбрать что-то, что будет вам очень нравиться и хорошо получаться — а в таких условиях стать профессионалом и достичь успеха легко.

Тестировщик «виноват во всём», т.е. с него спрос за все ошибки

Только если признать, что в болезни пациента виновен термометр, показывающий высокую температуру. Скорее с тестировщиков будет спрос за те ошибки, что были найдены пользователем, т.е. проявились уже на стадии реальной эксплуатации продукта. Но и здесь нет однозначного вывода — за конечный успех продукта отвечает вся команда, и было бы глупо перекладывать ответственность лишь на одну её часть.

Тестировщики скоро будут не нужны, т.к. всё будет автоматизировано

Как только по улицам забегают терминаторы — да, этот миф станет правдой: программы научатся обходиться без людей. Но тогда у нас всех будут другие проблемы. А если кроме шуток, человечество уже сотни лет идёт по пути автоматизации, которая накладывает свой отпечаток на всю нашу жизнь и чаще всего позволяет переложить самую простую и не требующую квалификации работу на машины. Но кто же заставляет вас оставаться на уровне исполнителя такой работы? Начиная с некоторого уровня, тестирование превращается в гармоничное сочетание науки и искусства. А многих ли учёных или творцов заменила автоматизация?

Просьба: возможно, у вас есть какие-то мысли из разряда «А я думал(а), что в тестировании…» / «А правда ли, что в тестировании…». Если так, поделитесь ими, пожалуйста, в анонимном опросе: http://svyatoslav.biz/software_testing_book_poll/