

# ПОДРОБНЫЙ ГАЙД ДЛЯ QA-СПЕЦИАЛИСТОВ

---

*От обучения до первой работы: конкретные шаги*

## СОДЕРЖАНИЕ

---

1. Что конкретно нужно изучить
2. Конкретные ресурсы для обучения
3. Пошаговый план обучения
4. Создание реального портфолио
5. Конкретные стратегии поиска работы
6. Подготовка к собеседованию на примерах

## 1. ЧТО КОНКРЕТНО НУЖНО ИЗУЧИТЬ

---

### Основы тестирования (обязательно):

- **Виды тестирования:** функциональное, регрессионное, smoke, интеграционное, UI/UX
- **Техники тест-дизайна:** классы эквивалентности, граничные значения, попарное тестирование
- **Тестовая документация:** создание чек-листов, тест-кейсов, баг-репортов
- **Жизненный цикл бага:** от обнаружения до закрытия

### Технические знания (необходимый минимум):

- **HTTP/HTTPS:** методы (GET, POST, PUT, DELETE), коды ответов, заголовки
- **Клиент-серверная архитектура:** как работают веб-приложения
- **Основы HTML/CSS:** структура страницы, селекторы (для локаторов при автоматизации)

- **Инструменты разработчика в браузере:** Chrome DevTools (Network, Console, Elements)
- **Базы данных:** базовый SQL (SELECT, INSERT, UPDATE, DELETE)
- **REST API:** принципы работы, тестирование с помощью Postman

### Инструменты (освоить практически):

- **Баг-трекинг:** Jira (создание задач, workflow, фильтры)
- **Тест-менеджмент:** TestRail или Qase (организация тест-планов и прогонов)
- **API-тестирование:** Postman (коллекции, окружения, тесты)
- **Системы контроля версий:** Git (базовые команды: clone, pull, push, branch)

## 2. КОНКРЕТНЫЕ РЕСУРСЫ ДЛЯ ОБУЧЕНИЯ

---

### Бесплатные ресурсы:

- **Курс "Тестирование ПО с нуля":** YouTube-канал Вадима Ксендзова
- **Основы тестирования:** Stepik курс "Введение в тестирование"
- **Тестирование веб-приложений:** [testingchallenges.testerswork.com](https://testingchallenges.testerswork.com)
- **Тестирование API:** Курс "API-тестирование с Postman" на YouTube
- **Git для тестировщиков:** Документация Git
- **SQL для тестировщиков:** Mode SQL Tutorial

### Платные курсы с высоким рейтингом:

- **QA-программа от SkillFactory:** полный курс с трудоустройством (от 40 000 ₽)
- **Яндекс.Практикум:** курс "Инженер по тестированию" (от 60 000 ₽)
- **Нетология:** курс "Тестировщик ПО" (от 45 000 ₽)
- **QA.Guru:** курс автоматизации на Java с нуля (от 50 000 ₽)

### Книги с практическими примерами:

- **"Тестирование Дот Ком" (Роман Савин)** - практические примеры и задачи

- "Тестирование черного ящика" (Борис Бейзер) - техники тестирования
- "Искусство тестирования программ" (Гленфорд Майерс) - фундаментальные принципы

### Практика на реальных проектах:

- **Crowdtesting платформы:** uTest, Тестбери (платят за найденные баги)
- **Open source проекты:** GitHub - список проектов для начинающих тестировщиков
- **Учебные приложения:** The Internet, Демо-магазин

## 3. ПОШАГОВЫЙ ПЛАН ОБУЧЕНИЯ (12 НЕДЕЛЬ)

### Неделя 1-2: Основы тестирования

- **День 1-3:** Изучите терминологию и типы тестирования (2-3 часа в день)
- **День 4-5:** Изучите техники тест-дизайна (2-3 часа в день)
- **День 6-7:** Научитесь создавать тестовую документацию (2-3 часа в день)
- **Практика:** Создайте 10 тест-кейсов для любимого приложения

### Неделя 3-4: Технические основы

- **День 1-3:** Изучите HTTP и клиент-серверную архитектуру (2-3 часа в день)
- **День 4-5:** Освойте основы HTML/CSS и DevTools (2-3 часа в день)
- **День 6-7:** Изучите основы Git и терминал (2-3 часа в день)
- **Практика:** Проанализируйте сетевые запросы любимого сайта через DevTools

### Неделя 5-6: Инструменты тестировщика

- **День 1-3:** Освойте Jira (создайте бесплатный аккаунт) (2-3 часа в день)
- **День 4-5:** Изучите TestRail или аналог (2-3 часа в день)
- **День 6-7:** Установите и настройте Postman (2-3 часа в день)
- **Практика:** Создайте проект в Jira и заведите 5 тестовых багов

## Неделя 7-8: API-тестирование

- **День 1-3:** Изучите принципы REST API (2-3 часа в день)
- **День 4-5:** Освойте тестирование API в Postman (2-3 часа в день)
- **День 6-7:** Научитесь писать тесты в Postman (2-3 часа в день)
- **Практика:** Протестируйте публичное API (например, JSONPlaceholder)

## Неделя 9-10: Базы данных и SQL

- **День 1-3:** Изучите основы реляционных БД (2-3 часа в день)
- **День 4-7:** Освойте основные SQL-запросы (2-3 часа в день)
- **Практика:** Установите SQLite и создайте простую базу для тестирования

## Неделя 11-12: Портфолио и подготовка к трудоустройству

- **День 1-3:** Создайте структуру для вашего портфолио (2-3 часа в день)
- **День 4-5:** Подготовьте резюме и профиль LinkedIn (2-3 часа в день)
- **День 6-7:** Изучите типичные вопросы на собеседования (2-3 часа в день)
- **Практика:** Запишите видео с демонстрацией вашего тестирования

# 4. СОЗДАНИЕ РЕАЛЬНОГО ПОРТФОЛИО

## Обязательные элементы портфолио:

- **Репозиторий на GitHub** с примерами вашей работы
- **Чек-листы для 2-3 реальных приложений** (минимум 20 пунктов в каждом)
- **30+ тест-кейсов** разных типов (позитивные, негативные, граничные значения)
- **15+ детальных баг-репортов** с скриншотами и видео воспроизведения
- **Коллекция API-тестов в Postman** (минимум 20 запросов с тестами)

## Пошаговое создание портфолио:

1. **Выберите 2-3 приложения для тестирования:**
  - Приложение для заметок <https://todoist.com/>
  - Интернет-магазин <https://demoqa.com/books>

- Каталог фильмов <https://www.themoviedb.org/>

## 2. Создайте тестовую документацию:

- Составьте требования (если их нет)
- Создайте чек-листы в Google Sheets
- Напишите тест-кейсы в TestRail (или в Markdown на GitHub)

## 3. Проведите тестирование:

- Выполните все тест-кейсы
- ЗадOCUMENTИРУЙТЕ найденные баги
- Запишите короткие видео воспроизведения с помощью Loom

## Структура репозитория на GitHub:

```
qa-portfolio/  
├─ app1-todoist/  
│  ├─ requirements.md  
│  ├─ checklist.md  
│  ├─ test-cases.md  
│  ├─ bug-reports/  
│  │  ├─ bug-001.md  
│  │  └─ ...  
│  └─ screenshots/  
├─ app2-demoqa/  
│  └─ ...  
├─ api-testing/  
│  └─ postman-collection.json  
└─ README.md
```

## 5. КОНКРЕТНЫЕ СТРАТЕГИИ ПОИСКА РАБОТЫ

### Подготовка документов (за 2 недели до поиска):

- **Резюме в формате ATS** (1-2 страницы максимум):
  - Используйте [резюме.hh.ru](https://резюме.hh.ru) для создания
  - Включите ключевые слова: QA Engineer, тестирование, Jira, Postman, SQL
  - Опишите ваши учебные проекты как реальный опыт работы
- **Сопроводительное письмо-шаблон** (3 абзаца):
  - Почему вас интересует QA

- Какие навыки у вас есть + ссылка на портфолио
- Почему вы хотите работать именно в этой компании

### Конкретные площадки для поиска:

- **HeadHunter:** создайте автопоиск по запросу "QA тестировщик стажер junior"
- **LinkedIn:** подпишитесь на хештеги #qaengineer #qajobs #testingjobs
- **Telegram-каналы:**
  - @qa\_jobs
  - @jobSoftQA
  - @qaoffers

### Тактика массовой рассылки (для первой работы):

1. Подготовьте список из 100+ компаний, нанимающих джуниоров
2. Отправляйте 10-15 резюме ежедневно
3. Настройте таблицу отслеживания в Google Sheets
4. Делайте follow-up через 3-5 дней после отправки резюме
5. Персонализируйте каждое письмо, упоминая продукт компании

### Пример таблицы отслеживания:

Компания	Вакансия	Дата отправки	Статус	Follow-up	Интервью	Результат
Company1	QA Intern	15.05.2025	Отказ	20.05.2025	-	Отказ

## 6. ПОДГОТОВКА К СОБЕСЕДОВАНИЮ НА ПРИМЕРАХ

### Топ-5 вопросов с примерами ответов:

1. **Вопрос:** Что такое тестирование ПО?

**Ответ:** "Это процесс проверки соответствия фактических результатов работы программы ожидаемым. Цель тестирования — не доказать, что продукт работает правильно, а найти как можно больше ошибок."

2. **Вопрос:** Какие виды тестирования вы знаете?

**Ответ:** "Функциональное (проверка функций), нефункциональное (производительность, безопасность), структурное (белый ящик).  
Конкретные примеры: smoke-тестирование, регрессионное, интеграционное, E2E, нагрузочное."

3. **Вопрос:** Чем отличается верификация от валидации?

**Ответ:** "Верификация отвечает на вопрос 'Правильно ли мы создаем продукт?' (соответствие спецификации). Валидация отвечает на вопрос 'Создаем ли мы правильный продукт?' (соответствие потребностям пользователя)."

4. **Вопрос:** Опишите хороший баг-репорт.

**Ответ:** "Хороший баг-репорт содержит: уникальный ID, краткое название, серьезность, приоритет, окружение, шаги воспроизведения, ожидаемый результат, фактический результат, приложения (скриншоты, видео, логи)."

5. **Вопрос:** Как бы вы протестировали калькулятор?

**Ответ:** "Сначала я бы разделил тестирование на части: UI, функциональность, граничные условия. Для функциональности проверил бы основные операции с разными типами чисел. Особое внимание уделил бы делению на ноль, переполнению разрядов, точности операций с плавающей точкой."

## Реальный тестовый кейс для демонстрации:

**Тест-кейс:** Проверка функциональности входа в систему

**Предусловия:**

- Существует учетная запись `test@example.com` с паролем `Test123!`
- Система доступна по адресу `https://example.com`

**Шаги:**

1. Открыть браузер
2. Перейти на страницу `https://example.com/login`

3. Ввести в поле "Email" значение test@example.com
4. Ввести в поле "Пароль" значение Test123!
5. Нажать кнопку "Войти"

**Ожидаемый результат:**

- Пользователь успешно авторизован
- Система перенаправляет на страницу личного кабинета
- В правом верхнем углу отображается имя пользователя

**Постусловия:**

- Выйти из системы через меню "Профиль" -> "Выйти"

Подписывайтесь на канал [@girl\\_fromthenorth](#) в Instagram для дополнительных материалов по QA и IT-карьере!