1-Week QA Freshers Training Plan

This weekly plan is designed for newcomers to QA. Each day includes bite-sized learning objectives, clear tasks, easy examples, and why each step matters. Follow this guide, ask your trainer if stuck, and make notes!

Day 1: Understanding Software Testing & QA Basics

- **What to Learn:**
  - What software testing is and why it matters
  - Difference between QA and QC
  - Role of a QA in a software project
- **Task:**
  - List 3 reasons software testing improves apps
  - Note down differences between QA & QC in a table
- **Examples:**
  - Think: "Testers find issues before users do" or "QA ensures documentation is followed"
- **Why it's Important:**
  - Knowing the basics helps you understand where your work fits in the software team

---

Day 2: Testing Methodologies & Life Cycle

- **What to Learn:**
  - Manual vs. automated testing (what's the difference)
  - Popular methodologies (Waterfall, Agile)
  - Software Testing Life Cycle (STLC) phases
- **Task:**
  - Draw a simple timeline of the Testing Life Cycle
  - Find key benefit for both manual and automation testing
- **Examples:**
  - Waterfall: sequential; Agile: fast feedback, flexible
  - STLC includes phases like planning, case design, execution
- **Why it's Important:**
  - Understanding the process helps plan your work and know what's expected at each stage

---

Day 3: Creating and Managing Test Cases

- **What to Learn:**
  - How to write clear, focused test cases
  - Parts of a test case: title, steps, expected result, actual result
- **Task:**
  - Write 3 test cases for a login page
  - Review examples with your teammate or trainer
- **Examples:**
  - "Verify login succeeds with correct username/password"
  - "Check error shown when password is missing"
- **Why it's Important:**
  - Well-written test cases guide testing, avoid missing critical scenarios, and are reusable

---

Day 4: Bug Reporting & Defect Life Cycle

- **What to Learn:**
  - How to describe and log a bug
  - Bug report fields (summary, steps, actual/expected behavior, severity)
  - Defect life cycle (New→Assigned→Fixed→Retested→Closed)
- **Task:**
  - Find a bug in a demo app, write a sample bug report in a tool like Jira or Trello
  - Label severity and priority
- **Examples:**
  - "Login button shows error: 'Server not found'"
  - Bug is set to 'Open', then 'In Progress', then 'Resolved'
- **Why it's Important:**
  - Clear bug reports let developers fix issues fast, and tracking ensures nothing is missed

---

Day 5: Introduction to Automation Testing

- **What to Learn:**
  - What automation is in testing, common tools (Selenium, Postman)
  - Basic advantages & limits of automation
- **Task:**
  - Watch a demo video of a Selenium test script
  - Open Selenium IDE and try running a sample test
  - How to find locators/elements?
- **Examples:**
  - Automating login tests for faster repeated checks
  - Record a simple browser test
- **Why it's Important:**
  - Automation helps test many scenarios quickly, reduces manual mistakes

---

Day 6: Performance Testing Introduction

- **What to Learn:**
  - What performance testing means (speed, reliability, load)
  - Tools used (Google Lighthouse, JMeter)
- **Task:**
  - Run Lighthouse on any website and note performance score
  - Discuss why slow sites frustrate users
- **Examples:**
  - "Site loads in 2 seconds vs. 10 seconds – which is better?"
  - What steps did Lighthouse suggest?
- **Why it's Important:**
  - Performance ensures users have a smooth experience, so issues are fixed before launch

---

Day 7: Recap, Real-World Scenarios & Q&A

- **What to Learn:**
  - Review all concepts learned; discuss how these fit together in a project
  - Common challenges in QA and how to address them
- **Task:**
  - Participate in a mock project: write a mini test plan, log 1 bug, automate one simple check
  - Share experiences/questions with team

- **Examples:**
  - "What would you test first on a shopping site?"
  - "How do you decide bug priority?"
- **Why it's Important:**
  - Practice ties knowledge together; sharing helps cement concepts and boost confidence

---

Helpful Resources & Tools for Beginners

- Websites:
  - BrowserStack Manual Testing Tutorial
  - SoftwareTestingHelp Live Project Free Training
- Tools to Try:
  - Test case writing: [Google Sheets, Excel]
  - Bug tracking: [Jira, Trello]
  - Automation: [Selenium IDE]
  - Performance: [Google Lighthouse]

Extra Tips for Freshers

- Ask questions if anything is confusing: senior team members want to help!
- Take notes and review them daily.
- Try practicing tasks on free demo apps/websites.
- Discuss your findings and mistakes in daily standups/check-ins.
- Try group exercises – learning from peers is powerful.

Remember: Every step gets you closer to being a skilled QA engineer. Stay curious, be hands-on, and take the week seriously. Your questions and learning now will pay off throughout your career!