Code Review – 100+ Lab Exercises (Basic, Intermediate, Advanced)

☐ Basic Level (30+ Exercises)

Objective: Build fundamental code review skills, understand syntax, readability, and simple refactorings.

Code Comprehension

- Review and summarize what a simple function does (in Python, JavaScript, or Java).
- Spot unused variables or unreachable code.
- Identify hardcoded values and suggest constant usage.
- Suggest better naming conventions for variables and functions.
- Explain the control flow of basic algorithms (e.g., sorting, searching).

Style & Readability

- Enforce consistent indentation and formatting (tabs vs spaces).
- Ensure consistent casing (camelCase, PascalCase, snake_case).
- Check alignment with style guides (e.g., PEP8, Google Java Style).
- Review HTML/CSS for structure and semantic tag usage.
- Identify comments that need clarification or removal.

Basic Best Practices

- Spot functions doing too many things (violating SRP).
- Suggest converting a loop into a more readable `map`/`filter`.
- Replace magic numbers with named constants.
- Remove redundant `else` blocks after `return`.
- Identify off-by-one errors in loops.

☐ Intermediate Level (40+ Exercises)

Objective: Review code for maintainability, performance, and adherence to design principles.

Code Efficiency & Complexity

- Analyze Big-O time/space complexity of a function.
- Refactor nested loops into more efficient solutions.
- Identify bottlenecks in recursive solutions.
- Suggest caching/memoization where applicable.
- Recommend use of built-in libraries or idiomatic syntax.

Design Patterns & Architecture

- Identify violations of SOLID principles.
- Review factory and singleton pattern implementations.
- Check for tight coupling and recommend dependency injection.
- Suggest breaking down monolithic classes into modules.
- Enforce separation of concerns (MVC, MVVM, etc.).

Testability & Unit Tests

- Review test cases for edge conditions.
- Ensure meaningful assertions in tests.
- Suggest mocking/stubbing over external API calls.
- Spot untested branches of code.
- Recommend parameterized tests and coverage tools.

Version Control Hygiene

- Identify oversized pull requests and suggest splitting.
- Review commit messages for clarity and atomicity.
- Ensure feature branches follow naming conventions.
- Spot unwanted debug logs or commented code.
- Enforce `.gitignore` compliance.

☐ Advanced Level (40+ Exercises)

Objective: Master code review across systems, security, scalability, and mentor others in the review process.

Security & Compliance

- Detect SQL injection vulnerabilities in raw queries.
- Review code for XSS, CSRF vulnerabilities in web apps.
- Validate authentication and token management flows.

- Check for usage of insecure hashing/encryption (e.g., MD5, SHA-1).
- Review 3rd-party library usage and CVE reports.

Concurrency & Scalability

- Review thread-safe implementation using locks, mutexes, or channels.
- Check async/await logic and race conditions.
- Validate scalability of microservices in distributed systems.
- Analyze rate limiting and load balancing logic.
- Review circuit breakers and retry patterns.

API & Integration

- Review RESTful API for status code accuracy.
- Validate GraphQL resolver design for N+1 query issues.
- Review API contract versioning and backward compatibility.
- Analyze response time optimization (pagination, compression).
- Ensure robust error handling and logging in API endpoints.

Mentorship & Collaboration

- Write constructive code review comments.
- Conduct mock live code review sessions.
- Mentor junior developers through example-based feedback.
- Drive cultural shift toward collaborative reviews.
- Champion best practices via review templates/checklists.

Capstone Review Projects

- Perform a full-stack application codebase review.
- Review a microservice end-to-end: infra, logic, and tests.
- Evaluate a monorepo for modularization and refactoring.
- Run a security audit on a payment/finance module.
- Simulate a review for an open-source pull request.

☐ Tools & Platforms

- **Linting & Formatters:** ESLint, Prettier, Black, Flake8
- **Code Review Platforms:** GitHub, GitLab, Bitbucket, Gerrit
- **CI Tools:** GitHub Actions, CircleCI, Jenkins

- **Testing Frameworks:** Jest, PyTest, JUnit, Mocha
- **Security Tools:** SonarQube, Snyk, CodeQL
- **Coverage Tools:** Istanbul, Codecov, JaCoCo
- **Architecture Review Aids:** PlantUML, Mermaid.js, Lucidchart