⭐ Code Review Rubric

Project Name: _____ **Author**: _____

Review Date: _____ **Reviewer**: _____

Aspect of Code Quality	Needs Improvement	Approaching Mastery	Professional
Readability & Formatting *Variable naming and casing* *Line length and complexity* *Formatting and indentation* *Explanations in comments*	☐ Unclear/arbitrary variable names ☐ Casing is sometimes inconsistent ☐ Lines are often long and complex ☐ Inconsistent formatting/indentation ☐ Few or no comments to explain complex or confusing code	☐ Descriptive variable names ☐ Casing is always consistent ☐ Lines are usually short and terse ☐ Readable formatting/indentation ☐ Several comments to explain complex or confusing code	☐ Clear, semantic variable names ☐ Casing always follows conventions ☐ Lines are always short and terse ☐ Consistent formatting/indentation ☐ Complex code is always explained with comments when appropriate
Organization & Modularity *Modularity and coupling* *Use of abstraction* *Side effects of functions*	☐ Code contains large monolithic or tightly coupled functions and/or classes that could be separated ☐ Limited or no use of abstraction ☐ Functions use global variables	☐ Code is separated into functions and/or classes but may be tightly coupled causing ripple of changes ☐ Some use of abstraction ☐ Few functions cause side effects	☐ Code is separated into functions and/or classes with different, clear responsibilities and loose coupling ☐ Abstraction used whenever helpful ☐ All functions avoid side effects
Effectiveness of Solution *Does it solve the problem?*	☐ Solves some typical input cases ☐ Does not solve any edge cases	☐ Solves most typical input cases ☐ Solves some obvious edge cases	☐ Solves all typical input cases ☐ Solves all known edge cases
Standard Library & Conventions *Uses existing functions/classes* *Follows language conventions*	☐ Several standard library functions or classes are reinvented without any customization or justification ☐ Violates language conventions	☐ Occasional use of standard library shows exposure and/or research ☐ Few cases of reinvention could be simplified using standard library	☐ Significant use of standard library when helpful and to simplify code and customizations are justifiable ☐ Follows language conventions
Testing & Error Handling *Testing solution robustness* *Handling errors and exceptions*	☐ Minimal or no automated testing ☐ Test inputs are simplistic or naive ☐ Minimal or no exception handling	☐ Tests cover typical input cases ☐ Test inputs are varied and creative ☐ Handles some errors/exceptions	☐ Tests cover all typical input cases ☐ Tests cover all known edge cases ☐ Handles several errors/exceptions
Algorithmic Complexity *Efficient use of resources* *Scalability with large inputs*	☐ Code often repeats redundant operations or uses brute force ☐ High algorithmic complexity that does not scale with large inputs	☐ Some code repeats redundant work, but with minimal impact ☐ Low algorithmic complexity that avoids brute force approaches	☐ Repeated work is often avoided to save time and memory resources ☐ Optimal algorithmic complexity that scales well with large inputs