Programming Rubric

Criteria	Beginning 50%	Developing 70%	Proficient 85%	Exemplary 100%
Code Functionality	Completed 0-69% of the requirements; program does not execute due to errors, and/or was not delivered on time.	Completed 70-89% of the requirements; program executes with minimal errors, and was delivered on time.	Completed 90-99% of the requirements; program executes without errors, and was delivered on time.	Completed 100% of the requirements; program executes without errors, and was delivered on time.
Adherence to Standards and Conventions	Poor use of indention, variable and function names, whitespace, semicolons, and organization. Does not meet standards and conventions leading to many errors.	Average use of indention, variable and function names, whitespace, semicolons, and organization is present and somewhat understandable. Standards and conventions has some errors.	Good use of indention, variable and function names, whitespace, semicolons, and organization. Meets standards and conventions with few errors.	Excellent use of indention, variable and function names, whitespace, semicolons, and organization. Meets standards and conventions without errors.
Efficiency and Use of Language Features	Poor use of functions, class objects, DRY (Don'tRepeatYourself) principles, and language features. Meets language features with many errors.	Use of functions, class objects, DRY (Don't-RepeatYourself) principles, and language features is present and somewhat identifiable. Meets language features with some errors.	Good use of functions, class objects, DRY (Don'tRepeatYourself) principles, and language features. Meets language features with few errors.	Excellent use of functions, class objects, DRY (Don'tRepeatYourself) principles, and language features. Meets language features without errors.
Documentation	Code is unorganized, poorly documented, comments have many errors in grammar, semantics, and spelling. Code attribution is not present and/or authorship is missing.	Code is somewhat neat, documented, and comments have some errors in grammar, semantics, and spelling. Code attribute is not present and/or authorship is not clearly annotated.	Code is neat, well documented, and code comments have few errors in grammar, semantics, and spelling. Code attribute is present and/or authorship is clearly annotated.	Code is neat, well documented, and code comments are free of grammar, semantics, and spelling errors. Code attribution is present and/or authorship is clearly annotated.
Error Trapping and Handling	Poor use of error handling. The code has several logical errors and exception messages have multiple errors in grammar, semantics, and spelling. Exceptions are not written to a log or output to the Console.	The program has some logical errors and exception messages have some errors in grammar, semantics, and spelling. Exceptions are not written to a log or output to the Console.	Good use of error handling. The code is free of logical errors and exception messages have few errors in grammar, semantics, and spelling. Caught exceptions are mostly written to a log or output to the Console.	Excellent use of error handling. The code is free of logical errors and exception messages are free of grammar, semantics, and spelling errors. Caught exceptions are either written to a log or output to the Console.
Assignment Specific Compliance	Complies with few or none of the assignment instructions and/or guidelines for format, citation, and style (e.g., APA). Many errors present.	Complies with some assignment instructions and/or guidelines for format, citation, and style (e.g., APA). Some errors present.	Complies with most assignment instructions and/or guidelines for format, citation, and style (e.g., APA). Few errors present.	Complies with all assignment instructions and/or guidelines for format, citation, and style (e.g., APA). No errors present.