OHD General Programming Standards and Guidelines Peer Review Checklist

Reviewer's Name:	Taylor Dudunake	**Peer Review Date:**	03/12/18
Project Name:	Jupyter Notebook Assignment 2	**Project ID:** Enter if applicable	
Developer's Name:	Gus Womeldorph	**Project Lead:**	
Review Files & Source code			
Code Approved			

This checklist is to be used to assess source code during a peer review. Items which represent the code being reviewed should be checked off.

Refer to the *OHD General Programming Standards and Guidelines* document for more complete descriptions and examples of the items listed below.

1. Internal Documentation

__✓__ Comment block exists at the beginning of the source file containing at least the following information: original author's name, file creation date, development group, and a brief statement of the purpose of the software in the file.

____ Each subroutine or function in the file is preceded by a comment block which provides the following information: routine name, original author's name, routine's creation date, purpose of the routine, a list of the calling arguments

(their types and what they do), a list of required files and/or database tables, the routine's return value, error codes and exceptions processed by the routine, and a modification history indicating when and by whom changes were made.

2. Programming Standards

N/A Consistent indentation of at least 3 spaces is used to distinguish conditional or control blocks of code. TABS NOT USED FOR INDENTATION.

____ Inline comments are frequently used and adequately describe the code.

✓ Structured programming techniques are adhered to.

✓ Subroutines, functions, and methods are reasonably sized.

✓ The routines in each source file shall have a common purpose or have interrelated functionality. Methods in a class support its functionality.

✓ The name of the file, script or class represents its function.

✓ Function and method names contain a verb, that is, they indicate an action.

✓ Meaningful variable names are used.

✓ Variables are initialized prior to use.

N/A Braces are used consistently

✓ There are no compiler warnings

3. Programming Guidelines

✓ Source file line lengths are 80 characters or less.

✓ Spacing is used correctly to enhance the source code's readability.

N/A When continuing lines of code on new lines, they are broken after a comma or an operator. Higher level breaks are used instead of lower level breaks.

N/A Wrapped lines of code are aligned with the beginning of the expression at the same level on the previous line.

N/A Multiple line variable declarations are preceded by a type.

✓ Program statements are limited to one per line.

✓ Nested program statements are avoided.

✓ Parentheses are used to remove operator precedence ambiguity and to improve code readability.

___ Inline comments constitute approximately 20% of the total lines of code in the program, excluding the file and routine documentation blocks.

✓ The software reflects a balance of coding for efficiency and coding for readability.

N/A Meaningful error messages are used.

N/A System calls which acquire system resources are tested for error returns.

✓ Routines and methods are contain no more than 200 executable lines.

✓ The number of routines in a source file is kept to a minimum for procedural languages.

Reviewer's Comments:

Overall, your code is well written and very simple; it is not extremely "calculation heavy". I don't have a problem reading your code as it's also very similar to mine in it's process. If I was you, I'd add a bit more comment blocks using a #-sign to ensure that the reader fully understands what is being done. Also, be sure to revise your comments (Task 7, specifically). Your figures are easy to read and comprehend, so good job on that. It demonstrates your attention to making sure the analysis is easy to follow. Well done, Gus!