COGS18: Project Grading Rubric

Overview

- In general, we are not aiming for particularly harsh grading. If they spent time and effort engaging in the project, they should get a passing grade.
- Sections aren't all or none - unless they totally didn't do something, give them some proportion of the section, matching the extent to which they did or didn't do the required thing.
- Keep in mind the students' context. Though you might know a better way to do something, it is not necessarily fair to expect them to know that. Keep the course materials in mind in terms of evaluating what they should know how to do.

Concept (total: 5 points)

5 points: They chose a project topic from either the assignment list, or an independent topic they can write code for

File Structure (total: 5 points)

1.5 points There is at least on Jupyter notebook file

1.5 points There is at least one python file

2 points Files are interpretable organized (not necessarily same as template)

Project Description (total: 10 points)

2.5 points There is a project description in a Jupyter notebook

7.5 points The project description is a self-contained explanation of the project. It provides a description of what they are trying to do.

Approach (total: 20 points)

10 points The design they chose responds to the project topic they outlined

10 points The code / approaches / algorithms they used for their project are appropriate for the task they trying to complete

Code (total: 30 points)

5 Variables & Code Constructs: they use variables and control flow constructs in their code, as needed

5 Functions & Classes: code is organized into functions Use of classes is optional

5 Modular Organization: at least some code is organized into an external module, and this file is well constructed

5 Imports & Libraries: they import code from the module, and use external libraries if & when appropriate.

10 The code is functional - it does what it is supposed to, with no major errors or bugs, and works as intended.

Style (total: 10 points)

2.5 points	There is good code layout - spacing between code segments, etc
2.5 points	There is proper indentation and spacing
2.5 points	Names used are descriptive
2.5 points	Names follow the right naming conventions (snake_case, CamelCase)

Documentation (total: 10 points)

7 points	There are numpy format docstrings on functions & classes Needs: first line sentence, parameters & returns sections
3 points	There are (at least some) inline comments that explain the code

Tests (total: 10 points)

2.5 points	Has asserts that are used to test some property of the code
2.5 points	These are organized into a well formed test file
2.5 points	Tests are organized into well formed test functions
2.5 points	The tests execute