DS2500
Spring 2022
Homework Grading Policy

We have weekly homework assignments this semester, related to recent lecture material. You'll submit two components each week:

1. A self-reflection, answering questions about how the assignment went and what you learned.
2. The programming assignment itself, usually .py or .ipynb files.

Your self-reflection will be graded. Answering all questions is sufficient for full homework credit.

Your solution to the programming assignment will not be traditionally graded. In your self-reflection, you have the option to ask us for feedback, and we'll review and comment on any components that you like.

This policy applies only to weekly homework assignments; projects will be graded traditionally on an A-F scale.

Homework Credit
You must submit your self-reflection and programming assignment by the homework due date. For full credit, you must answer all questions in the self-reflection (they will be specified on each homework assignment).

We won't look at your programming submission by default, other than your self-reflection. If your self-reflection requests feedback on any part of your work, we'll provide it. You can also come to office hours to go over your approach and ask any clarifying questions.

Late Policy
You may submit your homework up to 48 hours late; you'll still receive full credit, but we won't provide feedback.

Feedback
The feedback we give, when requested, will help refine your work and give a sense of what makes for good programming practice among data scientists. Projects in this course *will* be graded traditionally, and part of our intention with feedback is to let you know what we're looking for.

Feedback will give notes along the following categories:

Category	**What We Expect**
Program Correctness	Program meets specifications of the assignment and generates the expected output. No errors or warnings are produced when we run the code.
Readability + Reusability	Variable and function names are clear and concise. Code is modular and functions are used appropriately.

	Code is clean, understandable, and well-organized.
Documentation	Comments are clear and frequent; test cases or unit tests are provided for functions when appropriate.
Visualizations	Plots and other visuals created convey insights you learned about the data. They contain labels and legends that explain the content. The scale used makes sense for the data.