Midterm Portfolio Learning Targets & Grade Expectations

# Learning Targets

The following learning targets have been covered in Weeks 1-5 of this course. When locating code that “fits” into these learning targets I would like for you to:

* find 1-2 **high quality** examples (quality > quantity)
* find examples that hit on **multiple** learning targets at the same time

For example, the code below could be used to demonstrate proficiency with:

* WD-2 – in two places (select() and starts\_with())!
* WD-7
* WD-4
* R-2 (tidy code)
* PE-4 (the |> is a “modern tool”)

ca\_childcare |>  
 select(study\_year,  
 mc\_infant,  
 mc\_toddler,  
 mc\_preschool,  
 region  
 ) |>  
 pivot\_longer(cols = starts\_with("mc\_"),   
 names\_to = "age",   
 values\_to = "median\_income"  
 ) |>   
 mutate(age = fct\_relevel(age,   
 "mc\_infant",  
 "mc\_toddler",  
 "mc\_preschool")  
 )

Working with Data

* **WD-1**: I can import data from a variety of formats (e.g., csv, xlsx, txt)
* **WD-2**: I can select necessary columns from a dataset
* **WD-3**: I can filter rows from a dataset for a variety of data types (e.g., numeric, factor, date, character)
* **WD-4**: I can modify existing variables and create new variables in a dataset for a variety of data types (e.g., numeric, factor, date, character)
* **WD-5**: I can use mutating joins from the tidyr package to combine multiple datasets
* **WD-6**: I can use filtering joins from the dplyr package to filter rows from a dataset
* **WD-7**: I can pivot datasets from long to wide and visa versa

Reproducibility

* **R-1**: I can create professional looking, reproducible analyses using RStudio projects, Quarto documents, and the here package
* **R-2**: I can write well documented[[1]](#footnote-20) and tidy code[[2]](#footnote-21)
* **R-3**: I can write robust programs[[3]](#footnote-22) that are resistant to changes in inputs

Data Visualization & Summarization

* **DVS-1**: I can create visualizations for a variety of variable types (e.g., numeric, character, date)
* **DVS-2**: I use plot modifications to make my visualizations clearer to the reader
* **DVS-3**: I show creativity in my visualizations
* **DVS-4**: I can calculate numerical summaries of variables
* **DVS-5**: I can calculate numerical summaries of variables across multiple groups
* **DVS-6**: I can create tables which make my summaries clear to the reader

Program Efficiency

* **PE-1**: I can write concise code which does not repeat itself.
* **PE-4**: I can use modern tools[[4]](#footnote-23) when carrying out my analysis.

# Grade Guidelines

The following are the expectations for each letter grade, as written (and voted on) by you. When making an argument for your grade, your rationale should be based on these criteria rather than vague statements about the work you have done thus far (e.g., “I’d worked really hard each week to accomplish the assignments.”).

| Letter Grade | Description of Work and Participation |
| --- | --- |
| **A** (Superior Achievement) | **Learning Targets:** Meets the criteria for a B and also demonstrates proficiency with DVS-3, DVS-7, and PE-3, demonstrating proficiency with nearly every learning target.   **Revising Thinking:** Submits revisions for *all but a few* assignments, includes thoughtful reflections on why revisions are being requested, demonstrating an understanding of the “bigger picture” ideas. Takes all feedback into account (including problems receiving an S) and incorporates into future assignments.   **Extending Thinking:** *Consistently* making an effort to push oneself, attempting and succeeding at more difficult challenge problems, and incorporating what is learned into future assignments.   **Collaborative Group Member:** Is *always* patient and respectful when collaborating, makes a concerted effort to follow coder / developer roles, excellent communication, willing to discuss and listen, open to new ideas, not controlling.  **Peer-Code Review:** *Every week* provides specific, constructive feedback, which is honest and kind and demonstrates a desire to help others grow. |
| **B** (Good Achievement) | **Learning Targets:** Meets the criteria for a C and also demonstrates proficiency with objectives WD-5 through WD-7, DVS-2, PE-1, PE-2, PE-4, DSSM-1, and DSSM-2; effort is made toward DVS-3, DVS-7, and PE-3, but not at the level of proficiency  **Revising Thinking:** Submits revisions for *most* assignments (more than half), includes reflections on why revisions are being requested but reflections may miss some of the “bigger picture” ideas. These mistakes are fixed in future assignments, but additional feedback provided (on problems receiving an S) is not incorporated into future assignments.   **Extending Thinking:** *Sometimes* making an effort to push oneself, attempting and succeeding at more difficult challenge problems. *Sometimes* what is learned is incorporated into future assignments.   **Collaborative Group Member:** Is *always* patient and respectful when collaborating, excellent communication *mostly* follows coder / developer roles, but sometimes performs tasks outside their role.  **Peer-Code Review:** *Most* weeks provides specific, constructive feedback, which is honest and kind and demonstrates a desire to help others grow. |
| **C** (Acceptable Achievement) | **Learning Targets:** Demonstrates proficiency with objectives WD-1 through WD-4, R-1, R-2, DVS-4, DVS-5; slight effort is made toward WD-5 through WD-7 and PE-1 through PE-4, but not at the level of proficiency.   **Revising Thinking:** Submits revisions for some assignments (less than half), frequently fails to include reflections on why revisions are being requested. Mistakes continue to be made in future assignments, and additional feedback provided (on problems receiving an S) is not incorporated into future assignments.   **Extending Thinking:** Rarely making an effort to push oneself, mainly completing the easier challenges.   **Collaborative Group Member:** Is *mostly* respectful when collaborating, but *does not* follow coder / developer roles, making *little effort* to collaborate with their partner.  **Peer-Code Review:** *Rarely* provides feedback on peer’s assignments. When feedback is given, it is vague and does not offer concrete ways the person’s code formatting could be improved. |
| **D** | **Learning Targets:** Demonstrates proficiency with objectives WD-1 through WD-4, R-1, R-2, DVS-4, DVS-5. No effort is made toward additional learning targets.  **Revising Thinking:** Seldom if ever submits revisions. When revisions are submitted, reflections on why revisions are being requested are not included. Mistakes are not fixed in future assignments.   **Extending Thinking:** Seldom if ever makes an effort to push oneself, only completing the easier challenges.   **Collaborative Group Member:** Is *mostly* respectful when collaborating, but *does not* follow coder / developer roles, making *little effort* to collaborate with their partner.  **Peer-Code Review:** *Never* provides feedback on peer’s assignments. |
| **F** | **Learning Targets:** Does not satisfy the criteria for a D.  **Revising Thinking:** Does not satisfy the criteria for a D.   **Extending Thinking:** Does not satisfy the criteria for a D.   **Collaborative Group Member:** Does not satisfy the criteria for a D.   **Peer-Code Review:** Does not satisfy the criteria for a D. |

1. “well documented” code has elements of the following:

   code comments (e.g., # function to rescale columns between 0 and 1)

   code chunk names (using the #| lab: option)

   sections in Quarto file which describe the process(es) being implemented [↑](#footnote-ref-20)
2. “tidy code” is code which adheres to the tidyverse style guidelines [↑](#footnote-ref-21)
3. The follow are some examples of code I would consider “robust”:

   using column names rather than column positions

   including function stops when writing functions

   using processes that don’t depend on you remembering a particular sequence of steps (e.g., arrange() then slice() versus slice\_max()) [↑](#footnote-ref-22)
4. **The following are some examples of tools I would consider are “modern”:**

   functions which are not superseded

   functions which are not deprecated

   using the pipe operator to decrease the number of intermediate objects created [↑](#footnote-ref-23)