12 Loops+Iter P2

Learn to PURRR

1. What Purrr Is And Why It's Useful
    - purrr is the tidyverse's toolkit for functional iteration
    - replaces many of base-R's apply/loop patterns with simpler, cleaner functions
    - works especially well when you want to apply the same operation across many inputs without manually writing loops
2. map Functions As Core Tools
    - map() applies a function to each element of a list or vector and returns a new list
    - map_* variants let you return more specific output types (e.g. map_dbl(), map_df(), etc.) for easier downstream use
    - using the tilde-dot shorthand (~ .x^2, etc.) makes it concise to write simple anonymous functions inside map
3. Lists As Purrr's Workhorse
    - unlike base R vectors, lists can hold elements of any type (numbers, data frames, plots…)
    - purrr is especially powerful when working with lists of data frames (or other complex objects) — you can map a transformation or model across all of them with a few lines of code
4. Advanced Functional Patterns
    - you can map over multiple inputs simultaneously using map2() or pmap() when you need to iterate across multiple vectors or lists at once
    - purrr also provides tools for error-handling and safe execution across many operations (e.g. possibly(), safely(), quietly()) so that one failure doesn't crash the entire batch
5. Purrr + Tidyverse = Clean, Readable, Reproducible Code
    - using purrr keeps your code concise, consistent, and less prone to mistakes compared to manual loops or apply calls
    - makes workflows like "split-apply-combine" across grouped data or list-columns much easier and more elegant
    - helps avoid repetitive code when doing similar operations on many data pieces (e.g. multiple models, multiple plots, multiple data files)