11 Loops+Iter P1

26 Iteration

1. Understanding The Need For Iteration
   - iteration is required when you need to repeat an operation many times
   - copying and pasting code is inefficient and error-prone
   - loops and functional tools automate repeated tasks
2. For Loops
   - for loops run the same code for each element in a sequence
   - they are explicit, readable, and good for beginners
   - you must create an output object before filling it inside the loop
3. While Loops
   - while loops continue running as long as a condition remains true
   - they are useful when the total number of iterations is unknown
   - they require careful stopping conditions to avoid infinite loops
4. Breaking And Skipping
   - break exits a loop early
   - next skips to the next iteration without running the remaining code
   - these tools help control loop behavior
5. Using Purrr For Iteration
   - map() applies a function to each element of a vector or list
   - map_* variants simplify returning specific types like numeric or character
   - purrr functions are concise and integrate well with the tidyverse
6. Anonymous Functions
   - anonymous functions allow you to write small one-off functions inline
   - they are useful inside map() when the transformation is simple
   - using ~ syntax with purrr makes them shorter and cleaner
7. Iterating Over Multiple Inputs
   - map2() iterates over two vectors simultaneously
   - pmap() handles iterations over multiple inputs stored in lists or data frames
   - these functions simplify multi-argument iteration
8. Best Practices For Iteration
   - choose loops when clarity matters
   - choose purrr when working with tidyverse workflows
   - ensure your output structure is consistent and predictable
   - avoid deep nesting by breaking work into small helper functions