Lecture 5: Functions in R

In-Class Activities

1. Iterations

   (a) Create a `while()` loop that starts at `value <- 1` and repeatedly multiplies by `1.25` until the value is **at least** 100. Your loop should:

   - keep track of how many multiplications were needed (store this in `steps`)
   - store every intermediate value (including the starting value 1) in a vector called `history`
   - print `steps` and `history` after the `while()` loop has finished running.

   (b) Create a `for()` loop that prints your name exactly 10 times, **numbered** like this:

   ```
   1:  Dr.  McCurdy
   2:  Dr.  McCurdy
   ⋮
   10:  Dr.  McCurdy
   ```

2. Conditionals and Functions

   (a) Using the `mtcars` dataset, create a new column variable called `efficient` based on:

   - `"yes"` if the `mpg` is > 25
   - `"kinda"` if the `mpg` is > 20 but less than 25
   - `"no"` if the `mpg` is < 20

   (b) Create a function in R called `our_factorial(n)` that computes $n!$ using a `while()` loop and `return()`s the final value.

   Your function should:

   - handle `n = 0` correctly (should return 1). You may need to use the `ifelse()` function to specify this.
   - stop and return `NA` if `n` is negative or not a whole number. You may need to use the `if()` function along with the `break` argument to carry this out.

   (c) Create a function `ends_in_vowel(x)` that takes a character vector `x` and returns:

   - `"Vowel"` if *any* element ends in a vowel (a, e, i, o, u), ignoring case
   - `"No Vowel"` otherwise

   Requirements:

   - Use a `for()` loop to check each element.
   - Your function should **stop early** (break out of the loop) as soon as it finds one that ends in a vowel.

3. Apply / Aggregate

   (a) In the built-in `iris` dataset:

      i. Use `aggregate()` to compute the mean of `Sepal.Length` by `Species`.
      ii. Then do the same thing using `tapply()`.

   (b) In the `mlb_players_18` dataset (from the MSMU library):

      i. Use `aggregate()` to compute the mean of `HR` by `Position`.
      ii. Then use an `apply()`-family function to compute the mean of `HR` by `Position` again

   (c) Using `aggregate()` on `iris`, compute **both**:

      - mean `Petal.Length` by `Species`
      - max `Petal.Length` by `Species`

      Put your results into a single object (a dataframe is fine).

4. Put it all together

   (a) Create a function `print_name(name, n)` that prints `name` exactly `n` times.
      Requirements:

      - Use a `for()` loop.
      - Print each line as `i:  name` (so it is numbered).
      - If `n` is not a positive whole number, the function should return `NA` and print a helpful message.