Control Flow in R: Takeaways

Concepts

- Control flow is a form of decision making in code. Using comparison operators, we can decide what code to run based on if a value satisfies a given condition.
- The `if` statement provides us with a way to implement a *branching path* structure to our code. The `if_else()` and `case_when()` functions are vectorized implementations of the `if` statements, which we can use to create new columns in our dataset based on control flow.

Syntax

- An `if` statement can be written like the following:

```
if (insert comparison operator here) {
      print("Code to run if the comparison operator is TRUE")
    } else {
      print("Code to run if the comparison operator is FALSE")
    }
```

- The `if_else()` function vectorizes a simple two-branch decision tree:

```
new_recent_grads <- recent_grads %>%
      mutate(
    is_engineering = if_else(Major_category == "Engineering", TRUE, FALSE)
      )
```

- To create a multiple comparison in an `if_else()` function, you must use `&` and `|` instead of `&&` and `||`

- The `case_when()` function vectorizes a more complex, 2+ branch decision tree:

```
new_recent_grads <- recent_grads %>%
      mutate(
        size_classification = case_when(
          Total < 2000 ~ "Small",
          Total > 20000 ~ "Large",
          TRUE ~ "Medium
        )
      )
```

- The `%in%` operator helps us create a comparison operator based on a membership test. If a value is in a given collection, then it will evaluate to `TRUE`. `FALSE`, otherwise.

```
recent_grads %>%
      filter(
        Major %in% c("AEROSPACE ENGINEERING", "BIOMEDICAL ENGINEERING", "CHEMICAL ENGINEERING")
      )
```

- We can use the ! character to invert logical values and comparison operators. ` x <- 10
  x == 10 [1] TRUE
  !(x == 10) [1] FALSE `