Week-5: Code-along

Annette

2023-09-11

```
knitr::opts_chunk$set(echo = TRUE)
```

II. Code to edit and execute using the Code-along.Rmd file

A. Writing a function

1. Write a function to print a “Hello” message (Slide #14)

```
print("Hello")

## [1] "Hello"

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✔ dplyr     1.1.2     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.3     ✔ tibble    3.2.1
## ✔ lubridate 1.9.2     ✔ tidyr     1.3.0
## ✔ purrr     1.0.2
## ── Conflicts ──────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force
all conflicts to become errors
```

2. Function call with different input names (Slide #15)

```
say_hello_to <- function(name) {
      print(paste0("Hello ", name, "!"))
  }

say_hello_to("Annette")

## [1] "Hello Annette!"
```

3. typeof primitive functions (Slide #16)

```
typeof(`+`)

## [1] "builtin"

typeof(sum)

## [1] "builtin"
```

4. typeof user-defined functions (Slide #17)

```
typeof(say_hello_to)

## [1] "closure"

typeof(mean)

## [1] "closure"
```

5. Function to calculate mean of a sample (Slide #19)

```
calc_sample_mean <- function(sample_size) {
     mean(rnorm(sample_size))
 }
```

6. Test your function (Slide #22)

```
calc_sample_mean(1000)

## [1] 0.03721161
```

```
calc_sample_mean(c(100,300,3000))

## [1] 1.375996
```

7. Customizing the function to suit input (Slide #23)

```
sample_tibble <- tibble(sample_sizes = c(100,300,3000))

sample_tibble %>%
      group_by(sample_sizes) %>%
      mutate(sample_mean = calc_sample_mean(sample_sizes))

## # A tibble: 3 × 2
## # Groups:   sample_sizes [3]
##   sample_sizes sample_mean
##          <dbl>       <dbl>
## 1          100     0.00876
## 2          300     0.0826
## 3         3000     0.00648
```

8. Setting defaults (Slide #25)

```
calc_sample_mean <- function(sample_size, our_mean=0, our_sd=1) {
     sample <- rnorm(sample_size,
                     mean = our_mean,
                     sd= our_sd)
     mean(sample)
 }

calc_sample_mean(sample_size = 10)

## [1] 0.3550533
```

9. Different input combinations (Slide #26)

```
calc_sample_mean(10, our_mean = 6)

## [1] 6.020397
```

10. Different input combinations (Slide #27)

```
calc_sample_mean(our_mean = 5)

## Error in calc_sample_mean(our_mean = 5): argument "sample_size" is
missing, with no default
```

11. Some more examples (Slide #28)

```
add_two <- function(x) {
     x+2
}

add_two(43.3)

## [1] 45.3
```

B. Scoping

12. Multiple assignment of z (Slide #36)

```r
foo <- function( z =2) {
     z <- 3
     return(z+3)
 }
foo()

## [1] 6
```

13. Multiple assignment of z (Slide #37)

```r
foo <- function( z =2) {
     z <- 3
     return(z+3)
}
foo( z = 4)

## [1] 6
```