# ICA 1.2 Loading and addressing data

### Task - Object assignment

- 1. Supplement the code below so that the values are assigned into an object called height.
- 2. To see a representation of height, type its name on a line by itself and then run the line (Ctrl-Enter or \( \mathbb{K}\)-Enter). Try this now. print(obj\_name) (which "prints" to the screen) and show(obj\_name) do the same, where obj\_name is the object of interest.
- 3. Consider length(height). What do you think will be the result of this call? Add it to the chunk below and test your prediction.
- ☐ Tip: The keyboard shortcut for inserting the assignment operator is Alt (alt-hyphen) on Windows, or (option-hyphen) on Mac.

```
c(158.4, 176.8, 162.7, 169.6, 163.0, 165.3, 134.3, 190.7)

[1] 158.4 176.8 162.7 169.6 163.0 165.3 134.3 190.7

signif(rnorm(8, 161.3, .19*sqrt(5510)), 4)

[1] 142.6 177.2 160.0 173.1 169.7 152.6 166.1 159.6

# your code here
```

```
# 1
height <- c(158.4, 176.8, 162.7, 169.6, 163.0, 165.3, 134.3, 190.7)

# 2
height
# or:
print(height)

# 3
length(height)

The result is 8 because there are 8 values in the height vector.
```

# Task - Attach a package

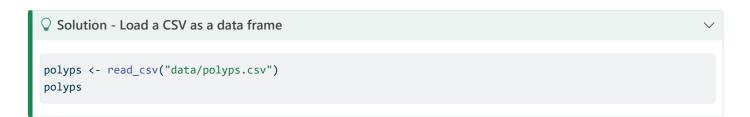
Let's attach the readr package to load the CSV file. Call library(readr) below.

Note that many packages have *dependencies*, meaning that they must attach and utilize other packages in order order to function.

#### Task - Load a CSV as a data frame

Let's load some data from a study which treated polyps.

- 1. Examine your file system (you can use the Files tab at bottom-right, or Explorer/Finder) for a directory within our project called data. Confirm that data contains a file called polyps.csv.
- 2. Use readr's read\_csv() function to load polyps.csv, and assign the result to an object called polyps.
- 3. Call the object's name to see a representation of it, and confirm you have loaded the file.
- # P
- # your code here



#### Task - A data frame's names

Try calling names() on your polyps data frame. What do you think this function does? Why might this be useful?

- # P
- # your code here



#### Task - Address a variable

- 1. Recall that \$ allows us to reference a variable in a data frame. For example, suppose we had an ecology dataset called penguins, with a variable called body\_mass\_g. To access the variable, we could type: penguins\$body\_mass\_g.
- 2. Use the \$ selector below with the information from names(polyps) to print the participant ages as a vector. (Hint: df\_name\$variable\_name is the format you want.)
- 3. Assign these values to a new object called ages.
- 4. Print ages to the screen.
- # P
- # your code here

```
Solution - Address a variable
polyps$age
ages <- polyps$age</p>
ages
```

#### Task - Arithmetic with variables

- 1. Calculate the mean of ages, rounded to two significant digits.
- 2. Calculate participant ages in months.
- 3. baseline and number3m are the counts of observed polyps *initially* (baseline), and after 3 months, respectively. Try subtracting baseline from number3m. What is this result?
- 4. Assign the subtraction result to a new object called diff\_3m. How many values does diff\_3m have? Why?
- 5. Print diff\_3m to the screen.
- # P
  # your code here

## Solution - Arithmetic with variables

```
# 1
signif(mean(ages), 2)
# 2
ages * 12
# 3
polyps$number3m - polyps$baseline
# 4
diff_3m <- polyps$number3m - polyps$baseline
length(diff_3m)
# 5
diff_3m</pre>
```

diff\_3m is each participant's reduction in polyps after 3 months. There are 22 values in the result because there were 22 values in the input, and subtraction is a vectorized operation.