**תרגיל 1.3**

**שאלה 1**

Write your own function to compute the variance of a numeric vector.

The corresponding function is:

variance <- **function**(x, na.rm = TRUE) {

n <- **length**(x)

m <- **mean**(x, na.rm = TRUE)

sq\_err <- (x - m)^2

**sum**(sq\_err) / (n - 1)

}

**var**(1:10)

*#> [1] 9.17*

**variance**(1:10)

*#> [1] 9.17*

**שאלה 2**

In the second variant of rescale01(), infinite values are left unchanged. Rewrite rescale01() so that -Inf is mapped to 0, and Inf is mapped to 1.

rescale01 <- **function**(x) {

rng <- **range**(x, na.rm = TRUE, finite = TRUE)

y <- (x - rng[1]) / (rng[2] - rng[1])

y[y == -Inf] <- 0

y[y == Inf] <- 1

y

}

**rescale01**(**c**(Inf, -Inf, 0:5, NA))

*#> [1] 1.0 0.0 0.0 0.2 0.4 0.6 0.8 1.0 NA*

**שאלה 3**

Write both\_na(), a function that takes two vectors of the same length and returns the number of positions that have an NA in both vectors.

both\_na <- **function**(x, y) {

**sum**(**is.na**(x) & **is.na**(y))

}

**both\_na**(

**c**(NA, NA, 1, 2),

**c**(NA, 1, NA, 2)

)

*#> [1] 1*

**both\_na**(

**c**(NA, NA, 1, 2, NA, NA, 1),

**c**(NA, 1, NA, 2, NA, NA, 1)

)

*#> [1] 3*

**תרגיל 2.1**

Write a greeting function that says “good morning”, “good afternoon”, or “good evening”, depending on the time of day. (Hint: use a time argument that defaults to lubridate::now(). That will make it easier to test your function.)

greet <- **function**(time = lubridate::**now**()) {

hr <- lubridate::**hour**(time)

*# I don't know what to do about times after midnight,*

*# are they evening or morning?*

**if** (hr < 12) {

**print**("good morning")

} **else** **if** (hr < 17) {

**print**("good afternoon")

} **else** {

**print**("good evening")

}

}

**greet**()

*#> [1] "good morning"*

**greet**(**ymd\_h**("2017-01-08:05"))

*#> [1] "good morning"*

**greet**(**ymd\_h**("2017-01-08:13"))

*#> [1] "good afternoon"*

**greet**(**ymd\_h**("2017-01-08:20"))

*#> [1] "good evening"*

**תרגיל 5.1**

**שאלה 1**

What does commas(letters, collapse = "-") do? Why?

The commas() function in the chapter is defined as

commas <- **function**(...) {

**str\_c**(..., collapse = ", ")

}

When commas() is given a collapse argument, it throws an error.

**commas**(letters, collapse = "-")

*#> Error in str\_c(..., collapse = ", "): formal argument "collapse" matched by multiple actual arguments*

This is because when the argument collapse is given to commas(), it is passed to str\_c() as part of .... In other words, the previous code is equivalent to

**str\_c**(letters, collapse = "-", collapse = ", ")

However, it is an error to give the same named argument to a function twice.

One way to allow the user to override the separator in commas() is to add a collapse argument to the function.

commas <- **function**(..., collapse = ", ") {

**str\_c**(..., collapse = collapse)

}

**שאלה 2**

What does the trim argument to mean() do? When might you use it?

The trim arguments trims a fraction of observations from each end of the vector (meaning the range) before calculating the mean. This is useful for calculating a measure of central tendency that is robust to outliers.

**שאלה 3**

The default value for the method argument to cor() is c("pearson", "kendall", "spearman"). What does that mean? What value is used by default?

It means that the method argument can take one of those three values. The first value, "pearson", is used by default.