# Libraries

library(readr)

# Arithmetic ——————————————

2+2

## [1] 4

3 + 3

## [1] 6

10 + 17 + 5

## [1] 32

5 - 3

## [1] 2

8 - 1

## [1] 7

5 \* 5

## [1] 25

2 \* 4

## [1] 8

1 / 2

## [1] 0.5

2 / 7

## [1] 0.2857143

2^8

## [1] 256

10^3

## [1] 1000

2\*\*8

## [1] 256

10\*\*3

## [1] 1000

2+3-6 / 2

## [1] 2

10 \* 2 - 3 / 5^2

## [1] 19.88

2 / (3 \* 2)

## [1] 0.3333333

(2 / 3) \* 2

## [1] 1.333333

-2 \* 3

## [1] -6

10 ^ -3

## [1] 0.001

# Variables and assignments —————————

x <- (12/3.5)^2 + (1/2.5)^3 + (1+2+3)^0.33  
x # returns 13.6254

## [1] 13.6254

y <- x^2  
y # returns 185.6516

## [1] 185.6516

y <- x \* 3.6  
y # returns 49.05145

## [1] 49.05145

# Vectors ———————————————-

primes <- c(2, 3, 5, 7, 11, 13, 17, 19, 23)  
primes + 1 # performs operation on every value in vector

## [1] 3 4 6 8 12 14 18 20 24

primes / 2

## [1] 1.0 1.5 2.5 3.5 5.5 6.5 8.5 9.5 11.5

primes ^ 2

## [1] 4 9 25 49 121 169 289 361 529

primes[1] # returns the first value in vector

## [1] 2

primes[5]

## [1] 11

primes[c(7,5,3)] # returns the 7th, 5th, and 3rd elements in primes

## [1] 17 11 5

primes[2:5] # returns 2nd, 3rd, 4th, and 5th elements of primes vector

## [1] 3 5 7 11

primes[-1] # returns all but the first element of primes

## [1] 3 5 7 11 13 17 19 23

primes[-2]

## [1] 2 5 7 11 13 17 19 23

primes[-c(7,5,3)] # returns all but the 7th, 5th, and 3rd elements of primes

## [1] 2 3 7 13 19 23

x[1] # single values are considered vectors with one element

## [1] 13.6254

class(primes) # returns "numeric"

## [1] "numeric"

nation <- c('ireland', 'england', 'scotland', 'wales', 'france', 'italy')  
class(nation) # returns "character"

## [1] "character"

nation[1] #will return 'ireland'

## [1] "ireland"

nation[-2] # returns all values except england

## [1] "ireland" "scotland" "wales" "france" "italy"

nation[4:6] #returns 4th, 5th, and 6th elements of nation

## [1] "wales" "france" "italy"

# nation + 2 # returns an error  
# nation \* 2 # returns an error  
  
is\_male <- c(TRUE, FALSE, TRUE, TRUE, FALSE)  
class(is\_male) # returns logical

## [1] "logical"

is\_male[2] # returns FALSE

## [1] FALSE

is\_male[c(1,2)] # returns TRUE FALSE

## [1] TRUE FALSE

result <- is\_male \* 2  
class(result) # returns "numeric"

## [1] "numeric"

TRUE & FALSE # returns FALSE

## [1] FALSE

TRUE & TRUE # returns TRUE

## [1] TRUE

c(T, F, T) & c(T, T, F) # returns TRUE FALSE FALSE

## [1] TRUE FALSE FALSE

TRUE | FALSE # return TRUE

## [1] TRUE

TRUE | TRUE # returns TRUE

## [1] TRUE

!TRUE #returns FALSE

## [1] FALSE

!FALSE #return TRUE

## [1] TRUE

(TRUE | !TRUE) & FALSE # returns FALSE

## [1] FALSE

(TRUE | !TRUE) & !FALSE # returns TRUE

## [1] TRUE

primes == 7 # returns logical value for each element in primes against the condition

## [1] FALSE FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE

12 == (6\*2)

## [1] TRUE

12 != (6\*2)

## [1] FALSE

primes != 3 # returns logical value for each element checking if it is not equal to 3

## [1] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

primes < 5

## [1] TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE

primes > 8

## [1] FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

primes <= 7

## [1] TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE

primes >= 3

## [1] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

nation == 'england'

## [1] FALSE TRUE FALSE FALSE FALSE FALSE

nation != 'england'

## [1] TRUE FALSE TRUE TRUE TRUE TRUE

nation >= 'italy'

## [1] FALSE FALSE TRUE TRUE FALSE TRUE

coerced\_num <- c(TRUE, FALSE, 3, 2, -1) #logical values will convert to numeric  
coerced\_str <- c(2.75, 11.3, TRUE, FALSE, 'dog', 'cat') #all values will become character strings  
  
c(primes, primes^2, primes^3)

## [1] 2 3 5 7 11 13 17 19 23 4 9 25  
## [13] 49 121 169 289 361 529 8 27 125 343 1331 2197  
## [25] 4913 6859 12167

ages <- c(bob = 27, bill = 19, charles = 24)  
ages

## bob bill charles   
## 27 19 24

ages['bob'] # returns 27

## bob   
## 27

ages['bill'] # returns 19

## bill   
## 19

ages\_again <- c(27, 19, 24)  
names(ages\_again) <- c('bob', 'bill', 'charles')  
ages\_again # equivalent to ages

## bob bill charles   
## 27 19 24

w <- c(1, 2, NA, 4, 5)  
w

## [1] 1 2 NA 4 5

z <- c('be', NA, 'afeard')  
z

## [1] "be" NA "afeard"

c(1, 2, NA, 4, 5) # numeric vector

## [1] 1 2 NA 4 5

c(1, 2, 'NA', 4, 5) # character vector

## [1] "1" "2" "NA" "4" "5"

# Data Frames —————————————

data\_df <- data.frame(name = c('billy', 'joe', 'bob'), age = c(21, 29, 23))  
data\_df[3,2] # returns value at row 3 column 2

## [1] 23

data\_df[c(1,3), 2] # returns values of 1st and 3rd row in 2nd column

## [1] 21 23

data\_df[3,] # returns all elements in third row

## name age  
## 3 bob 23

data\_df[,2] # returns all values of the 2nd column

## [1] 21 29 23

data\_df$age

## [1] 21 29 23

data\_df[['age']]

## [1] 21 29 23

data\_df['age'] #returns subset of data\_df that is itself a data frame

## age  
## 1 21  
## 2 29  
## 3 23

#Other data structures —————————–

example\_list <- list(A = TRUE, B = c(1, 2, 3), C = c('cat', 'dog'))  
example\_list

## $A  
## [1] TRUE  
##   
## $B  
## [1] 1 2 3  
##   
## $C  
## [1] "cat" "dog"

example\_list[['B']]

## [1] 1 2 3

example\_list$B

## [1] 1 2 3

example\_list[[1]]

## [1] TRUE

matrix(c(2, 3, 5, 7, 11, 13, 17, 19, 23, 29), nrow=2, ncol=5)

## [,1] [,2] [,3] [,4] [,5]  
## [1,] 2 5 11 17 23  
## [2,] 3 7 13 19 29

matrix(c(2, 3, 5, 7, 11, 13, 17, 19, 23, 29), byrow = T, nrow=2, ncol=5)

## [,1] [,2] [,3] [,4] [,5]  
## [1,] 2 3 5 7 11  
## [2,] 13 17 19 23 29

rbind(c(1, 2), c(3, 4)) # binds by row

## [,1] [,2]  
## [1,] 1 2  
## [2,] 3 4

cbind(c(1, 2), c(3, 4)) # binds by column

## [,1] [,2]  
## [1,] 1 3  
## [2,] 2 4

primes\_m <- matrix(c(2, 3, 5, 7, 11, 13, 17, 19, 23, 29), byrow = T, nrow=2, ncol=5)  
primes\_m

## [,1] [,2] [,3] [,4] [,5]  
## [1,] 2 3 5 7 11  
## [2,] 13 17 19 23 29

primes\_m[1, 3] # row 1, column 3

## [1] 5

primes\_m[2, ] # row 2, all cols

## [1] 13 17 19 23 29

primes\_m[, 3] # column 3, all rows

## [1] 5 19

example\_array <- array(c(1, 2, 3, 4, 5, 6, 7, 8), dim=c(2,2,2))  
example\_array[1,2,]

## [1] 3 7

example\_array[,1,]

## [,1] [,2]  
## [1,] 1 5  
## [2,] 2 6

example\_array

## , , 1  
##   
## [,1] [,2]  
## [1,] 1 3  
## [2,] 2 4  
##   
## , , 2  
##   
## [,1] [,2]  
## [1,] 5 7  
## [2,] 6 8

# Functions —————————————-

length(primes)

## [1] 9

sum(primes)

## [1] 100

mean(primes)

## [1] 11.11111

median(primes)

## [1] 11

sd(primes)

## [1] 7.457286

var(primes)

## [1] 55.61111

round(sqrt(mean(primes)))

## [1] 3

mean(primes, trim = 0.1) # trims the 10% highest and lowest values

## [1] 11.11111

my\_mean <- function(x){ sum(x)/length(x) } # custom function  
my\_mean(primes)

## [1] 11.11111

# Read File —————————————–

data <- read\_csv("C:\\Users\\janna\\OneDrive\\Documents\\DSE5001\\Week 1\\weight.csv")

## Rows: 6068 Columns: 8  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (1): gender  
## dbl (7): subjectid, height, height\_selfreport, weight, weight\_selfreport, ag...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.