Introduction to R: Core Language Tutorial

Dr Jeromy Anglim

```r
source("data-prep.R")
```

Basic Arithmetic and Logical Operations

```r
# You can use R like a calculator
1 + 1 # addition
```

```
## [1] 2
```

```r
10 - 9 # subtraction
```

```
## [1] 1
```

```r
10 * 10 # multiplcation
```

```
## [1] 100
```

```r
100 / 10 # division
```

```
## [1] 10
```

```r
10 ^ 2 # exponentiation
```

```
## [1] 100
```

```r
abs(-10) # absolute value
```

```
## [1] 10
```

```r
ceiling(3.5) # round up to next integer
```

```
## [1] 4
```

```r
floor(3.5) # round down to next integer
```

```
## [1] 3
```

```r
sqrt(100) # square roots
```

```
## [1] 10
```

```r
exp(2) # exponents
```

```
## [1] 7.389056
```

```r
pi # mathematical constant pi
```

```
## [1] 3.141593
```

```r
exp(1) # mathematical constant e
```

```
## [1] 2.718282
```

```r
log(100) # natural logs (i.e., base e)
```

```
## [1] 4.60517
```

```r
log(100, base= 10) # base 10 logs
```

```
## [1] 2
```

```r
# Use parentheses to clarify order of operations
(1 + 1 ) * 2
```

```
## [1] 4
```

```r
1 + (1 * 2)
```

```
## [1] 3
```

```r
# You can test for equality
# TRUE and FALSE are keywords
# T and F are synonyms, but are generally discouraged
TRUE
```

```
## [1] TRUE
```

```r
FALSE
```

```
## [1] FALSE
```

```r
1 == 2 # Equality (Return TRUE if equal)
```

```
## [1] FALSE
```

```r
1 != 2 # Inequality (Return FALSE if unequal)
```

```
## [1] TRUE
```

```r
10 > 9 # Greater than
```

```
## [1] TRUE
```

```r
9 < 10 # Less than
```

```
## [1] TRUE
```

```r
10 <= 10 # Less than or equal
```

```
## [1] TRUE
```

```r
2 %in% c(1, 2 ,3) # is the number in the vector
```

```
## [1] TRUE
```

```r
# TRUE and FALSE coerces to 1 and 0 respectively
as.numeric(TRUE)
```

```
## [1] 1
```

```r
as.numeric(FALSE)
```

```
## [1] 0
```

```r
# Logical converting to 0, 1 is useful
x <- c(2, 5 ,7 ,10, 15)
x > 5
```

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

```r
sum(x > 5)  # sum of a 0-1 variable is a count
```

```
## [1] 3
```

```r
mean(x > 5) # mean of a 0-1 variable is a proportion
```

```
## [1] 0.6
```

Basic language features

```r
########################################################
# Assignment:
# To assign values to a variable either use <- or =
# <- is the more common convention in R
x <- 1 + 1
x
```

```
## [1] 2
```

```r
# = is the common assignment operator in other programming
# languages. It does work in R, but is not the convention.
y = 1 + 1
y
```

```
## [1] 2
```

```r
########################################################
# Variable name rules:
# Variable names generally
# 1. Start with a letter (lower or uppercase)
# 2. Followed by letters, numbers, underscore (_), or period (.)
# 3. No spaces

# These do not work
# my variable <- 1234
# 1234variable <- 1234
# 1234 <- 1234

# This works
myvariable <- 1234
my_variable <- 1234
my_variable <- 1234
myvariable123 <- 1234
myVariable <- 1234
my.variable <- 1234

# R has many naming conventions
# As a matter of preference, style, and convenience, I prefer:
# 1. Short but descriptive names
#    * Less than 8 characters for names of lists and data.frames
#    * Less than 15 characters for variables names in data.frames
# 2. Use underscore to separate words within a variable name
# 3. Avoid upper case letters

# Names starting with a period are hidden
.myvariable <- 1234
ls()
```

```
## [1] "csurvey"       "my_variable"   "my.variable"   "myvariable"
## [5] "myVariable"    "myvariable123" "x"             "y"
```

```r
ls(all.names = TRUE)
```

```
##  [1] ".myvariable"   ".Random.seed"  "csurvey"       "my_variable"
##  [5] "my.variable"   "myvariable"    "myVariable"    "myvariable123"
##  [9] "x"             "y"
```

```r
#########################################################
# Comments:
# Comments are any text on a line following a hash #
# 1. They often appear as the first character of a line
#    to present a whole line comment
# 2. At the end of a common on a line
mean(c(1,2,3,4)) # Example of end of line comment
```

```
## [1] 2.5
```

```r
# 3. Half way through a command at the end of a line
c(1, # Example comment
  2,3,   # Another comment
  4)
```

```
## [1] 1 2 3 4
```

```r
#########################################################
# Spaces:
# R will generally permits zero, one or more spaces between
# variables, operators, and other syntactic elements.
# However, appropriate and consistent spacing improves
# the readability of you scripts.
# See Hadley Wickham's style guide:
# http://adv-r.had.co.nz/Style.html

# This is bad but works
x<-c(1,2,3,400)*2
x<-    c (   1,2,3,      400)*   2

# This is more readable:
# Add spaces after variables, operators, commas
x <- c(1, 2, 3, 400) * 2


########################################################
# Multipline line commands
# Commands can generally span multiple lines
# as long as R does not think the command has finished

# This works
x <- c("apple",
       "banana")
x
```

```
## [1] "apple"  "banana"
```

```r
y <- 10 +
    10  #this works
```

```
y
```

```
## [1] 20
```

```
# This does not work
y <- 10
    + 10
```

```
## [1] 10
```

```
y
```

```
## [1] 10
```

```
##########################################################
# Multiple commands on one line
# You can include more than one command on one line
# by separat the commands by a semicolon.
# But generally, you should avoid doing this as it is not
# very readable.
x <- c(1, 2); y  <- c(3, 4); z <- rnorm(10)
x;y;z
```

```
## [1] 1 2
```

```
## [1] 3 4
```

```
##  [1] -0.09585945  0.73305089  0.84742665 -0.38502517  0.21117101
##  [6] -0.71524025 -0.15603042 -2.00137618 -0.80437769 -0.71095037
```

```
##########################################################
# # R is case sensitive
test <- "lower case"
TEST <- "upper case"
TEST
```

```
## [1] "upper case"
```

```
test # The original value was not lost
```

```
## [1] "lower case"
```

```
     # because test is different to TEST
Test # This variable does not exist
```

```
## Error in eval(expr, envir, enclos): object 'Test' not found
```

```
Test <- "title case"
Test
```

```
## [1] "title case"
```

```
# tip: It's often simpler to make variables lower case
#      so that you don't have to think about case.
```

Understanding directories

```
# R has a working directory.
# This is important when loading and saving files to disk
getwd() # show the current working directory
```

```
## [1] "/Users/jeromy/teaching/r-training/acpid-2019-rtraining/materials/introduction-to-r/training-exe
```

```
# you can use setwd to change the working directory
# setwd("~/blah/myproject")

# Tip: Open RStudio with the Rproj file then the working directory
# will be the directory containing the Rproj file.

# Tips:
# * Try to avoid spaces in file names
#   (use hyphen or underscore instead)
# * If on Windows, then disable "hide extensions of
#   known file types" (see folder options )
# * If you do use spaces, then you'll need to escape the space with
#   a slash  (e.g., ("my\ documents")
# * Use backslash as the directory separator
# * Store all relevant files for a project within
#   the project working directory
```

The Workspace

```
###############################################
# R Sessions:

# Quitting R
# You can end the R Session using the q function
# q()

# But if you are in Rstudio, it is simpler to:
# * Just quit RStudio and this will quit the R session
# * Use the session menu in RStudio to Restart or Terminate
#   an R session


################################################
# Workspaces and environments:
# list environments
search()
```

```
##  [1] ".GlobalEnv"        "package:boot"      "package:metafor"
##  [4] "package:lavaan"    "package:lme4"      "package:Matrix"
##  [7] "package:dplyr"     "package:MASS"      "package:AER"
## [10] "package:sandwich"  "package:lmtest"    "package:zoo"
## [13] "package:car"       "package:carData"   "package:Hmisc"
## [16] "package:ggplot2"   "package:Formula"   "package:survival"
## [19] "package:lattice"   "package:psych"     "package:stats"
## [22] "package:graphics"  "package:grDevices" "package:utils"
## [25] "package:datasets"  "package:methods"   "Autoloads"
## [28] "package:base"
```

```
# Create some objects in the global environment
x <- 1:10
y <- 1:20
```

```r
data(mtcars) # Add a built-in datset mtcars

# Show objects in the global environment
ls()
```

```
##  [1] "csurvey"       "mtcars"        "my_variable"   "my.variable"
##  [5] "myvariable"    "myVariable"    "myvariable123" "test"
##  [9] "Test"          "TEST"          "x"             "y"
## [13] "z"
```

```r
# or look at the environment pane in RStudio

################################################
# Removing objects:
# Removing named objects with the rm function
rm(x)
ls()
```

```
##  [1] "csurvey"       "mtcars"        "my_variable"   "my.variable"
##  [5] "myvariable"    "myVariable"    "myvariable123" "test"
##  [9] "Test"          "TEST"          "y"             "z"
```

```r
rm(y, mtcars)

# Remove all objects from global workspace
# Option 1. Use the following command
rm(list = ls())
# Option 2. Click the broom object in RStudio Environment pane

###############################################
# Saving objects
# Save all objects in the workspace
# save.image()
save.image(file = "output/everything.rdata")

x <- 30
y <- 1:10
# Save specific named objects using save function.
# rdata or RData is the standard file exetnsion.
save(x, y, file = "output/y.rdata")

# Let's remove x and change y
rm(x)
y <- "changed"
y
```

```
## [1] "changed"
```

```r
# load variables stored in rdata file
load(file = "output/y.rdata")
x
```

```
## [1] 30
```

```r
y
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```r
# Tips:
# * Try to avoid using save.image() to store temporary calculations
# * Instead, try to write scripts that can be run to return you to
#   your current state of analyses.
```

Data types: Logical, character, numeric

```r
########################################################
# Basic data types
# The most common basic vector types are
x <- c(FALSE, TRUE) # logical vector
y <- c("a", "b", "cat", "dog") # character vector
z1 <- c(100, 1, 2, 3) # numeric integer vector
z2 <- c(100.2, 0.4, 0.9) # numeric real/double vector
class(x);  typeof(x); mode(x)
```

```
## [1] "logical"
## [1] "logical"
## [1] "logical"
```

```r
class(y);  typeof(y); mode(y)
```

```
## [1] "character"
## [1] "character"
## [1] "character"
```

```r
class(z1);  typeof(z1); mode(z1)
```

```
## [1] "numeric"
## [1] "double"
## [1] "numeric"
```

```r
class(z2);  typeof(z2); mode(z2)
```

```
## [1] "numeric"
## [1] "double"
## [1] "numeric"
```

```r
# Checking type of object
# there are a range of "is." functions for that return TRUE
# if object is of corresponding type
# apropos("^is\\.")
is.logical(c(TRUE, TRUE))
```

```
## [1] TRUE
```

```r
is.numeric(c("a", "b"))
```

```
## [1] FALSE
```

```r
is.character(c(1, 2, 3))
```

```
## [1] FALSE
```

```r
##############################################################
# Conversion of Types:
# R has functions that explicitly convert data types
# apropos("^as\\.")
as.character(c(1, 2, 3, 4))
```

```
## [1] "1" "2" "3" "4"
```

```r
as.numeric(c("1", "2a", "3", "four"))
```

```
## Warning: NAs introduced by coercion
## [1]  1 NA  3 NA
```

```r
as.numeric(c(FALSE, FALSE, TRUE, TRUE))
```

```
## [1] 0 0 1 1
```

```r
# R often performs conversions implicitly
sum(c(FALSE, TRUE, TRUE)) # converts logical to 0, 1 numeric
```

```
## [1] 2
```

```r
paste0("v", c(1, 2, 3)) # converts numeric vector to character
```

```
## [1] "v1" "v2" "v3"
```

Basic data structures: Vectors, Matrices, Lists, Data.frames

```r
############################################################
# Vectors:
# In R, a single value (scalar) is a vector.
x <- 1 # I.e., x is a vector of length 1


# In addition to importing data,
# R has various functions for creating vectors.
c(1, 2, 3, 4) # c stands for combine
```

```
## [1] 1 2 3 4
```

```r
1:10 # create an integer sequence 1 to 10
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```r
seq(1, 10) # alternative way of creating a sequence
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```r
seq(1, 10, by = 2) # The function has additional options
```

```
## [1] 1 3 5 7 9
```

```r
rep(1, 5) # repeat a value a certain number of times
```

```
## [1] 1 1 1 1 1
```

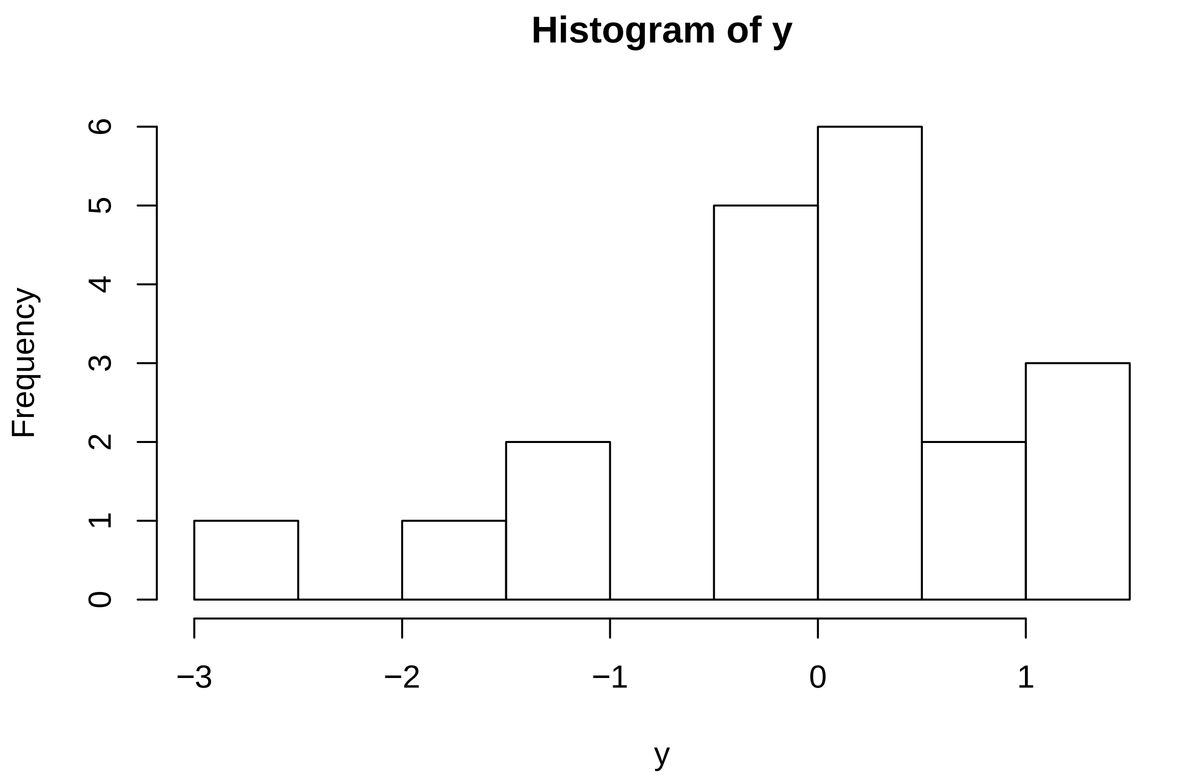
```
rep(c(1,2,3), 5) # repeat a value a certain number of times
```

```
##  [1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
```

```
# as well as many simulation functions which we'll cover later
# Initial examples:
# Sample 10 items with replacement from
sample(x = c("happy", "funny", "silly"), size = 10, replace = TRUE)
```

```
##  [1] "happy" "silly" "happy" "silly" "funny" "funny" "silly" "silly"
##  [9] "happy" "happy"
```

```
# Sample 20 values from a normal distribution
y <- rnorm(n = 20, mean = 0, sd = 1)
hist(y) # show values in histogram
```




```
# Vectors can have names
x <- c(1,2,3,4,5)
names(x) <- c("a", "b", "c", "d", "e")
x
```

```
## a b c d e
## 1 2 3 4 5
```

```
# Extracting vectors
x[c(1,2)] # by numeric position
```

```
## a b
## 1 2
```

```
x[x < 3] # by logical vector
```

```
## a b
```

```
## 1 2
```

```r
x[c("b", "c")] # by name
```

```
## b c
## 2 3
```

```r
###########################################################
# Matrices:
# All data must be of same type (e.g., numeric, character, logical)
y <- matrix(c(1, 2,
              4, 5,
              7, 8 ),
            byrow = TRUE, ncol = 2)
y
```

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

```r
class(y)
```

```
## [1] "matrix"
```

```r
# number of rows and columns
dim(y)  # Number of rows and columns
```

```
## [1] 3 2
```

```r
nrow(y) # Number of rows
```

```
## [1] 3
```

```r
ncol(y) # Number of columns
```

```
## [1] 2
```

```r
# Rows and columns can be given names
rownames(y) <- c("a", "b", "c")
colnames(y) <- c("col1", "col2")

# Rows and columns can be indexed
y["a", ] # By rowname
```

```
## col1 col2
##    1    2
```

```r
y[, "col1"] # By column name
```

```
## a b c
## 1 4 7
```

```r
y["a", "col1"] # By both
```

```
## [1] 1
```

```r
y[c(1,2), ] # By row position
```

```
##   col1 col2
## a    1    2
```

```
## b    4    5
```

```r
y[,1] # By column position
```

```
## a b c
## 1 4 7
```

```r
y[c(2,3), 2] # By column position
```

```
## b c
## 5 8
```

```r
############################################################
# Lists
# Store arbitrary structures of one or more named elements.
# Elements can be of different lengths
# Lists can contain lists can be nested to create tree like structures
# Lists are commonly used for representing results of analyses

w <- list(apple = c("a", "b", "c"),
          banana = c(1,2),
          carrot = FALSE,
          animals = list(dog = c("dog1", "dog2"),
                         cat = c(TRUE, FALSE)))

class(w)
```

```
## [1] "list"
```

```r
# Accessing one element of list
w$apple # using dollar notation
```

```
## [1] "a" "b" "c"
```

```r
w[[1]] # by position
```

```
## [1] "a" "b" "c"
```

```r
w[["apple"]] # by name (double brackets)
```

```
## [1] "a" "b" "c"
```

```r
# Accessing subset of list
w[c(1, 2)]  # by position (single bracket)
```

```
## $apple
## [1] "a" "b" "c"
##
## $banana
## [1] 1 2
```

```r
w[c("apple", "banana")] # by name
```

```
## $apple
## [1] "a" "b" "c"
##
## $banana
## [1] 1 2
```

```r
w[c(FALSE, FALSE, TRUE, TRUE)] # by logical vector
```

```
## $carrot
## [1] FALSE
##
## $animals
## $animals$dog
## [1] "dog1" "dog2"
##
## $animals$cat
## [1]  TRUE FALSE
```

```
# Quick illustration of a list object returned by
# a statistical function

# We'll simulate some data for two hypothetical groups x and y
# and perform an independent samples t-test.
x <- rnorm(10, mean = 0, sd = 1)
y <- rnorm(10, mean = 1, sd = 1)
fit <- t.test(x, y)

# The function
class(fit) # class does not say list, but it is a list
```

```
## [1] "htest"
```

```
mode(fit)
```

```
## [1] "list"
```

```
str(fit) # show structure of object
```

```
## List of 9
##  $ statistic  : Named num -2.29
##   ..- attr(*, "names")= chr "t"
##  $ parameter  : Named num 18
##   ..- attr(*, "names")= chr "df"
##  $ p.value    : num 0.0341
##  $ conf.int   : num [1:2] -2.882 -0.126
##   ..- attr(*, "conf.level")= num 0.95
##  $ estimate   : Named num [1:2] -0.483 1.021
##   ..- attr(*, "names")= chr [1:2] "mean of x" "mean of y"
##  $ null.value : Named num 0
##   ..- attr(*, "names")= chr "difference in means"
##  $ alternative: chr "two.sided"
##  $ method     : chr "Welch Two Sample t-test"
##  $ data.name  : chr "x and y"
##  - attr(*, "class")= chr "htest"
```

```
names(fit) # show names of elements
```

```
## [1] "statistic"   "parameter"   "p.value"     "conf.int"    "estimate"
## [6] "null.value"  "alternative" "method"      "data.name"
```

```
# we can view particular elements
fit$statistic
```

```
##         t
## -2.292472
```

```r
fit$parameter
```

```
##       df
## 17.99844
```

```r
fit$p.value
```

```
## [1] 0.03414527
```

```r
# or extract subsets of the list
fit[c("statistic", "parameter", "p.value")]
```

```
## $statistic
##         t
## -2.292472
##
## $parameter
##       df
## 17.99844
##
## $p.value
## [1] 0.03414527
```

```r
###########################################################
# Data Frames:
# Data frames are the standard data strucure used for storing
# data. If you have used other software (e.g., SPSS, Excel, etc.),
# this is what you may think of as a "dataset".
# Columns can be of different data types (e.g., character, numeric, logical, etc.)
z <- data.frame(var1 = 1:9, var2 = letters[1:9])
z
```

```
##   var1 var2
## 1    1    a
## 2    2    b
## 3    3    c
## 4    4    d
## 5    5    e
## 6    6    f
## 7    7    g
## 8    8    h
## 9    9    i
```

```r
# Tip: Some functions work with matrices,
#   some work with data.frames,
#   and some work with both.
# * If you are wanting to store data like you might store in
#   a database, then you'll generaly want a data.frame.
# * If you are dealing with a mathematical object that you
#   you want to perform a mathematical operation on, then you generally
#   want a matrix (e.g., correlation matrix, covariance matrix,
#   distance matrix in MDS, matrices used for matrix algebra).
```

Working with data frames

```
# Let's use the built-in survey data.frame dataset
library(MASS)
data(survey)
?survey

mydata <- survey

###################################################
# Extracting observations (i.e., rows) and
# variables (i.e., columns).
# There are similarities to matrices and lists
# Select observations
mydata[1:5, ] # by row number
```

```
##      Sex Wr.Hnd NW.Hnd W.Hnd    Fold Pulse    Clap Exer Smoke Height
## 1 Female   18.5   18.0 Right  R on L    92    Left Some Never  173.0
## 2   Male   19.5   20.5  Left  R on L   104    Left None Regul  177.8
## 3   Male   18.0   13.3 Right  L on R    87 Neither None Occas     NA
## 4   Male   18.8   18.9 Right  R on L    NA Neither None Never  160.0
## 5   Male   20.0   20.0 Right Neither    35   Right Some Never  165.0
##        M.I    Age
## 1   Metric 18.250
## 2 Imperial 17.583
## 3     <NA> 16.917
## 4   Metric 20.333
## 5   Metric 23.667
```

```
mydata[c(5,4,3,2,1), ] # re-order
```

```
##      Sex Wr.Hnd NW.Hnd W.Hnd    Fold Pulse    Clap Exer Smoke Height
## 5   Male   20.0   20.0 Right Neither    35   Right Some Never  165.0
## 4   Male   18.8   18.9 Right  R on L    NA Neither None Never  160.0
## 3   Male   18.0   13.3 Right  L on R    87 Neither None Occas     NA
## 2   Male   19.5   20.5  Left  R on L   104    Left None Regul  177.8
## 1 Female   18.5   18.0 Right  R on L    92    Left Some Never  173.0
##        M.I    Age
## 5   Metric 23.667
## 4   Metric 20.333
## 3     <NA> 16.917
## 2 Imperial 17.583
## 1   Metric 18.250
```

```
mydata[ mydata$Sex == "Female", ] # by logical vector
```

```
##        Sex Wr.Hnd NW.Hnd W.Hnd    Fold Pulse    Clap Exer Smoke Height
## 1   Female   18.5   18.0 Right  R on L    92    Left Some Never 173.00
## 6   Female   18.0   17.7 Right  L on R    64   Right Some Never 172.72
## 8   Female   17.0   17.3 Right  R on L    74   Right Freq Never 157.00
## 11  Female   17.0   17.2 Right  L on R    80   Right Freq Never 156.20
## 13  Female   16.0   16.0 Right  L on R    NA   Right Some Never 155.00
## 14  Female   19.5   20.2 Right  L on R    66 Neither Some Never 155.00
## 16  Female   17.5   17.0 Right  R on L    NA   Right Freq Never 156.00
## 17  Female   18.0   18.0 Right  L on R    89 Neither Freq Never 157.00
```

```
## 25  Female   17.0   17.5 Right  R on L    64    Left Some Never     NA
## 31  Female   18.5   18.0 Right  R on L    76   Right None Occas     NA
## 33  Female   17.1   17.5 Right  R on L    72   Right Freq Heavy 166.40
## 37  Female   16.0   16.5 Right  L on R    NA   Right Some Never 168.00
## 41  Female   17.5   16.0 Right  L on R    NA   Right Some Never 169.00
## 42  Female   17.8   18.0 Right  R on L    72   Right Some Never 154.94
## 44  Female   20.1   20.2 Right  L on R    80   Right Some Never 176.50
## 45  Female   13.0   13.0  <NA>  L on R    70    Left Freq Never 180.34
## 49  Female   18.0   17.6 Right  R on L    60   Right Some Occas 168.00
## 50  Female   18.0   17.9 Right  R on L    50    Left None Never 165.00
## 57  Female   15.5   15.4 Right  R on L    70 Neither None Never 157.48
## 62  Female   18.5   18.2 Right  R on L    72 Neither Freq Never 167.64
## 63  Female   19.6   19.7 Right  L on R    70   Right Freq Never 178.00
## 64  Female   18.7   18.0  Left  L on R    NA    Left None Never 170.00
## 65  Female   17.3   18.0 Right  L on R    64 Neither Freq Never 164.00
## 67  Female   19.0   19.1 Right  L on R    NA Neither Freq Never 172.00
## 68  Female   18.5   18.0 Right  R on L    64   Right Freq Never     NA
## 71  Female   18.0   17.5 Right  L on R    64    Left Freq Never 170.00
## 73  Female   17.0   16.6 Right  R on L    68   Right Some Never 171.00
## 74  Female   16.5   17.0 Right  L on R    40    Left Freq Never 167.64
## 75  Female   15.6   15.8 Right  R on L    88    Left Some Never 165.00
## 76  Female   17.5   17.5 Right Neither    68   Right Freq Heavy 170.00
## 77  Female   17.0   17.6 Right  L on R    76   Right Some Never 165.00
## 78  Female   18.6   18.0 Right  L on R    NA Neither Freq Heavy 165.10
## 79  Female   18.3   18.5 Right  R on L    68 Neither Some Never 165.10
## 83  Female   17.5   17.5 Right  R on L    98    Left Freq Never     NA
## 84  Female   17.0   17.4 Right  R on L    NA Neither Some Never     NA
## 86  Female   17.7   17.0 Right  R on L    76   Right Some Never 167.00
## 87  Female   18.2   18.0 Right  L on R    70   Right Some Never 162.56
## 88  Female   18.3   18.5 Right  R on L    75    Left Freq Never 170.00
## 90  Female   18.0   17.7  Left  R on L    92    Left Some Never     NA
## 92  Female   17.5   18.0 Right Neither    NA   Right Some Never     NA
## 93  Female   18.2   17.5 Right  L on R    70   Right Some Never 165.00
## 94  Female   18.2   18.5 Right  R on L    NA   Right Some Never 168.00
## 96  Female   19.0   18.8 Right  L on R    NA   Right Some Never     NA
## 98  Female   17.5   17.5 Right  R on L    60   Right Freq Never 166.50
## 100 Female   19.4   19.6 Right  R on L    68 Neither Freq Never 175.26
## 103 Female   16.0   16.0 Right Neither    NA   Right Some Never 159.00
## 104 Female   17.5   17.3 Right  R on L    72   Right Freq Never 175.00
## 105 Female   17.5   17.0 Right  R on L    80    Left Some Heavy 163.00
## 106 Female   19.5   18.5 Right  R on L    80   Right Some Never 170.00
## 107 Female   16.2   16.4 Right  R on L    NA   Right Freq Occas 172.00
## 108 Female   17.0   15.9 Right  R on L    85   Right Freq Never     NA
## 111 Female   18.5   18.5 Right  R on L    76    Left Freq Never 175.00
## 113 Female   17.2   16.7 Right  R on L    75   Right Freq Never 170.18
## 115 Female   16.0   15.5 Right  L on R    60    Left Freq Never 162.56
## 116 Female   16.9   16.0 Right  L on R    70   Right None Never 158.00
## 117 Female   17.0   16.7 Right  R on L    70   Right Some Never 159.00
## 119 Female   18.5   18.0  Left  L on R   100 Neither Some Never 171.00
## 123 Female   18.5   18.0 Right  R on L    92   Right Freq Never 172.00
## 127 Female   16.0   16.0 Right  R on L    68   Right Freq Never 172.72
## 129 Female   17.5   17.0 Right  R on L    74   Right Freq Never 157.00
## 130 Female   16.4   16.5 Right  L on R    90   Right Some Never 152.00
## 133 Female   18.9   20.0 Right  R on L    86   Right Some Never     NA
```

```
## 134 Female   15.4   16.4  Left  L on R    80    Left Freq Occas 160.02
## NA    <NA>     NA     NA  <NA>    <NA>    NA    <NA> <NA>  <NA>     NA
## 140 Female   19.5   18.5 Right  L on R    68   Right None Never 167.00
## 141 Female   18.0   18.6 Right  R on L    84   Right Some Never 175.00
## 142 Female   18.3   19.0 Right  R on L    NA   Right None Never 165.00
## 143 Female   19.0   18.8 Right  R on L    65   Right Freq Never 172.72
## 145 Female   20.0   19.5  Left  R on L    68 Neither Freq Never 172.00
## 149 Female   18.0   18.0 Right  L on R    92 Neither Freq Never 165.00
## 150 Female   18.0   18.5 Right  R on L    64 Neither Freq Never 164.00
## 152 Female   13.0   12.5 Right  L on R    80   Right Freq Never 165.00
## 153 Female   16.3   16.2 Right  L on R    92   Right Some Regul 152.40
## 158 Female   18.9   19.2 Right  L on R    74   Right Some Never 167.64
## 161 Female   17.5   17.1 Right  R on L    80    Left None Never 167.00
## 164 Female   16.5   16.9 Right  R on L    60 Neither Freq Occas 169.20
## 166 Female   17.6   17.2 Right  R on L    81    Left Some Never 168.00
## 167 Female   19.5   19.2 Right  R on L    70   Right Some Never 170.00
## 168 Female   16.5   15.0 Right  L on R    65   Right Some Regul 160.02
## 171 Female   16.5   17.0 Right  L on R    NA   Right Some Never 168.00
## 173 Female   15.5   15.5 Right Neither    50   Right Some Regul     NA
## 174 Female   18.0   17.5 Right  R on L    48 Neither Freq Never 165.00
## 175 Female   17.5   18.0 Right  R on L    68 Neither Freq Never 157.48
## 176 Female   19.0   18.5  Left  L on R   104    Left Freq Never 170.00
## 178 Female   16.7   17.0 Right  L on R    84    Left Freq Never 164.00
## 179 Female   20.5   20.5 Right  R on L    NA    Left Freq Regul     NA
## 180 Female   17.0   16.5 Right  R on L    70   Right Some Never 162.56
## 182 Female   14.0   13.5 Right  R on L    87 Neither Freq Occas 165.10
## 183 Female   17.5   17.6 Right  L on R    79   Right Some Never 162.50
## 187 Female   17.0   17.0 Right  L on R    79   Right Some Never 163.00
## 194 Female   17.6   17.8 Right  L on R    72    Left Some Never 160.02
## 195 Female   16.7   15.1 Right Neither    NA   Right None Never 157.48
## 196 Female   17.0   17.6 Right  L on R    76   Right Some Never 165.00
## 197 Female   15.0   13.0 Right  R on L    80 Neither Freq Never 170.18
## 199 Female   19.1   19.0 Right  R on L    80   Right Some Occas 170.00
## 200 Female   17.5   16.5 Right  R on L    80 Neither Some Never 164.00
## 201 Female   16.2   15.8 Right  R on L    61   Right Some Occas 167.00
## 203 Female   18.8   17.8 Right  R on L    76   Right Some Never     NA
## 204 Female   18.5   18.0 Right Neither    86   Right None Never 160.00
## 206 Female   17.5   17.0 Right  R on L    83 Neither Freq Occas 168.00
## 207 Female   17.5   17.6 Right  L on R    76   Right Some Never 153.50
## 210 Female   20.8   20.7 Right  R on L    NA Neither Freq Never 171.50
## 211 Female   18.6   18.6 Right  L on R    74   Right Some Never 160.00
## 212 Female   17.5   17.5  Left  R on L    83 Neither Some Never 163.00
## 215 Female   18.0   17.8 Right  L on R    68   Right Some Never 168.90
## 217 Female   16.3   16.2 Right  L on R    NA   Right None Never     NA
## 219 Female   17.0   17.3 Right  L on R    NA Neither Freq Never 173.00
## 222 Female   15.9   16.5 Right  R on L    70   Right Freq Never 167.64
## 223 Female   17.5   18.4 Right  R on L    88   Right Some Never 162.56
## 224 Female   17.5   17.6 Right  L on R    NA   Right Freq Never 150.00
## 225 Female   17.6   17.2 Right  L on R    NA   Right Some Never     NA
## 226 Female   17.5   17.8 Right  R on L    96   Right Some Never     NA
## 227 Female   18.8   18.3 Right  R on L    80   Right Some Heavy 170.18
## 229 Female   18.6   18.8 Right  L on R    70   Right Freq Regul 167.00
## 231 Female   18.8   18.5 Right  R on L    80   Right Some Never 169.00
## 233 Female   18.0   18.0 Right  L on R    85   Right Some Never 165.10
```

```
## 234 Female   18.5   18.0 Right  L on R    88   Right Some Never 160.00
## 235 Female   17.5   16.5 Right  R on L    NA   Right Some Never 170.00
## 237 Female   17.6   17.3 Right  R on L    85   Right Freq Never 168.50
##           M.I    Age
## 1     Metric 18.250
## 6   Imperial 21.000
## 8     Metric 35.833
## 11  Imperial 28.500
## 13    Metric 18.750
## 14    Metric 17.500
## 16    Metric 17.167
## 17    Metric 19.333
## 25      <NA> 19.167
## 31      <NA> 41.583
## 33  Imperial 39.750
## 37    Metric 19.000
## 41    Metric 17.500
## 42  Imperial 17.083
## 44  Imperial 17.500
## 45  Imperial 17.417
## 49    Metric 18.417
## 50    Metric 30.750
## 57  Imperial 17.167
## 62  Imperial 17.333
## 63    Metric 17.500
## 64    Metric 19.833
## 65    Metric 18.583
## 67    Metric 30.667
## 68      <NA> 16.917
## 71    Metric 17.583
## 73    Metric 17.667
## 74  Imperial 17.417
## 75    Metric 17.750
## 76    Metric 20.667
## 77    Metric 23.583
## 78  Imperial 17.167
## 79  Imperial 17.083
## 83      <NA> 17.667
## 84      <NA> 17.167
## 86    Metric 17.250
## 87  Imperial 18.000
## 88    Metric 18.750
## 90      <NA> 17.583
## 92      <NA> 18.000
## 93    Metric 19.667
## 94    Metric 17.083
## 96      <NA> 17.083
## 98    Metric 23.250
## 100 Imperial 19.083
## 103   Metric 20.833
## 104   Metric 20.167
## 105   Metric 17.667
## 106   Metric 18.250
## 107   Metric 17.000
```

```
## 108     <NA> 18.500
## 111   Metric 24.167
## 113 Imperial 21.167
## 115 Imperial 17.417
## 116   Metric 20.500
## 117   Metric 22.917
## 119   Metric 18.917
## 123   Metric 17.500
## 127 Imperial 17.667
## 129   Metric 18.000
## 130   Metric 18.333
## 133     <NA> 19.083
## 134 Imperial 18.500
## NA      <NA>     NA
## 140   Metric 18.667
## 141   Metric 17.500
## 142   Metric 21.083
## 143 Imperial 17.250
## 145   Metric 19.167
## 149   Metric 20.000
## 150   Metric 20.167
## 152   Metric 18.167
## 153 Imperial 23.500
## 158 Imperial 44.250
## 161   Metric 18.417
## 164   Metric 29.083
## 166   Metric 18.500
## 167   Metric 18.167
## 168 Imperial 32.750
## 171   Metric 73.000
## 173     <NA> 18.500
## 174   Metric 18.667
## 175 Imperial 17.750
## 176   Metric 17.250
## 178   Metric 23.083
## 179     <NA> 19.250
## 180 Imperial 17.167
## 182 Imperial 17.083
## 183   Metric 17.250
## 187   Metric 24.667
## 194 Imperial 17.250
## 195 Imperial 18.167
## 196   Metric 26.500
## 197 Imperial 17.000
## 199   Metric 19.167
## 200   Metric 17.500
## 201   Metric 19.250
## 203     <NA> 18.583
## 204   Metric 20.167
## 206   Metric 17.083
## 207   Metric 17.417
## 210   Metric 18.500
## 211   Metric 17.167
## 212   Metric 17.250
```

```
## 215 Imperial 17.083
## 217     <NA> 19.250
## 219   Metric 19.167
## 222 Imperial 17.333
## 223 Imperial 18.167
## 224   Metric 20.750
## 225     <NA> 19.917
## 226     <NA> 18.667
## 227 Imperial 18.417
## 229   Metric 20.333
## 231   Metric 18.167
## 233 Imperial 17.667
## 234   Metric 16.917
## 235   Metric 18.583
## 237   Metric 17.750
```

```
mydata[c("1", "2"), ] # by rownames
```

```
##      Sex Wr.Hnd NW.Hnd W.Hnd   Fold Pulse Clap Exer Smoke Height      M.I
## 1 Female   18.5   18.0 Right R on L    92 Left Some Never  173.0   Metric
## 2   Male   19.5   20.5  Left R on L   104 Left None Regul  177.8 Imperial
##      Age
## 1 18.250
## 2 17.583
```

```
# Select variables
mydata[, c(1,2)] # by position like a matrix
```

```
##        Sex Wr.Hnd
## 1   Female   18.5
## 2     Male   19.5
## 3     Male   18.0
## 4     Male   18.8
## 5     Male   20.0
## 6   Female   18.0
## 7     Male   17.7
## 8   Female   17.0
## 9     Male   20.0
## 10    Male   18.5
## 11  Female   17.0
## 12    Male   21.0
## 13  Female   16.0
## 14  Female   19.5
## 15    Male   16.0
## 16  Female   17.5
## 17  Female   18.0
## 18    Male   19.4
## 19    Male   20.5
## 20    Male   21.0
## 21    Male   21.5
## 22    Male   20.1
## 23    Male   18.5
## 24    Male   21.5
## 25  Female   17.0
## 26    Male   18.5
```

```
## 27    Male   21.0
## 28    Male   20.8
## 29    Male   17.8
## 30    Male   19.5
## 31  Female   18.5
## 32    Male   18.8
## 33  Female   17.1
## 34    Male   20.1
## 35    Male   18.0
## 36    Male   22.2
## 37  Female   16.0
## 38    Male   19.4
## 39    Male   22.0
## 40    Male   19.0
## 41  Female   17.5
## 42  Female   17.8
## 43    Male     NA
## 44  Female   20.1
## 45  Female   13.0
## 46    Male   17.0
## 47    Male   23.2
## 48    Male   22.5
## 49  Female   18.0
## 50  Female   18.0
## 51    Male   22.0
## 52    Male   20.5
## 53    Male   17.0
## 54    Male   20.5
## 55    Male   22.5
## 56    Male   18.5
## 57  Female   15.5
## 58    Male   19.5
## 59    Male   19.5
## 60    Male   20.6
## 61    Male   22.8
## 62  Female   18.5
## 63  Female   19.6
## 64  Female   18.7
## 65  Female   17.3
## 66    Male   19.5
## 67  Female   19.0
## 68  Female   18.5
## 69    Male   19.0
## 70    Male   21.0
## 71  Female   18.0
## 72    Male   19.4
## 73  Female   17.0
## 74  Female   16.5
## 75  Female   15.6
## 76  Female   17.5
## 77  Female   17.0
## 78  Female   18.6
## 79  Female   18.3
## 80    Male   20.0
```

```
## 81    Male   19.5
## 82    Male   19.2
## 83  Female   17.5
## 84  Female   17.0
## 85    Male   23.0
## 86  Female   17.7
## 87  Female   18.2
## 88  Female   18.3
## 89    Male   18.0
## 90  Female   18.0
## 91    Male   20.5
## 92  Female   17.5
## 93  Female   18.2
## 94  Female   18.2
## 95    Male   21.3
## 96  Female   19.0
## 97    Male   20.0
## 98  Female   17.5
## 99    Male   19.5
## 100 Female   19.4
## 101   Male   21.9
## 102   Male   18.9
## 103 Female   16.0
## 104 Female   17.5
## 105 Female   17.5
## 106 Female   19.5
## 107 Female   16.2
## 108 Female   17.0
## 109   Male   17.5
## 110   Male   19.7
## 111 Female   18.5
## 112   Male   19.2
## 113 Female   17.2
## 114   Male   20.5
## 115 Female   16.0
## 116 Female   16.9
## 117 Female   17.0
## 118   Male   23.0
## 119 Female   18.5
## 120   Male   21.0
## 121   Male   20.0
## 122   Male   22.5
## 123 Female   18.5
## 124   Male   19.8
## 125   Male   18.5
## 126   Male   19.3
## 127 Female   16.0
## 128   Male   18.8
## 129 Female   17.5
## 130 Female   16.4
## 131   Male   22.0
## 132   Male   19.0
## 133 Female   18.9
## 134 Female   15.4
```

```
## 135   Male   17.9
## 136   Male   23.1
## 137   <NA>   19.8
## 138   Male   22.0
## 139   Male   20.0
## 140 Female   19.5
## 141 Female   18.0
## 142 Female   18.3
## 143 Female   19.0
## 144   Male   21.4
## 145 Female   20.0
## 146   Male   18.5
## 147   Male   22.5
## 148   Male   19.5
## 149 Female   18.0
## 150 Female   18.0
## 151   Male   21.8
## 152 Female   13.0
## 153 Female   16.3
## 154   Male   21.5
## 155   Male   18.9
## 156   Male   20.5
## 157   Male   14.0
## 158 Female   18.9
## 159   Male   20.0
## 160   Male   18.5
## 161 Female   17.5
## 162   Male   18.1
## 163   Male   20.2
## 164 Female   16.5
## 165   Male   19.1
## 166 Female   17.6
## 167 Female   19.5
## 168 Female   16.5
## 169   Male   19.0
## 170   Male   19.0
## 171 Female   16.5
## 172   Male   20.5
## 173 Female   15.5
## 174 Female   18.0
## 175 Female   17.5
## 176 Female   19.0
## 177   Male   20.5
## 178 Female   16.7
## 179 Female   20.5
## 180 Female   17.0
## 181   Male   19.0
## 182 Female   14.0
## 183 Female   17.5
## 184   Male   18.5
## 185   Male   18.0
## 186   Male   20.5
## 187 Female   17.0
## 188   Male   18.5
```

```
## 189   Male   18.0
## 190   Male   18.5
## 191   Male   20.0
## 192   Male   22.0
## 193   Male   17.9
## 194 Female   17.6
## 195 Female   16.7
## 196 Female   17.0
## 197 Female   15.0
## 198   Male   16.0
## 199 Female   19.1
## 200 Female   17.5
## 201 Female   16.2
## 202   Male   21.0
## 203 Female   18.8
## 204 Female   18.5
## 205   Male   17.0
## 206 Female   17.5
## 207 Female   17.5
## 208   Male   17.5
## 209   Male   17.5
## 210 Female   20.8
## 211 Female   18.6
## 212 Female   17.5
## 213   Male   18.0
## 214   Male   17.0
## 215 Female   18.0
## 216   Male   19.5
## 217 Female   16.3
## 218   Male   18.2
## 219 Female   17.0
## 220   Male   23.2
## 221   Male   23.2
## 222 Female   15.9
## 223 Female   17.5
## 224 Female   17.5
## 225 Female   17.6
## 226 Female   17.5
## 227 Female   18.8
## 228   Male   20.0
## 229 Female   18.6
## 230   Male   18.6
## 231 Female   18.8
## 232   Male   18.0
## 233 Female   18.0
## 234 Female   18.5
## 235 Female   17.5
## 236   Male   21.0
## 237 Female   17.6
```

```
mydata[c(1,2)] # by position like a list
```

```
##        Sex Wr.Hnd
## 1   Female   18.5
## 2     Male   19.5
```

```
## 3     Male   18.0
## 4     Male   18.8
## 5     Male   20.0
## 6   Female   18.0
## 7     Male   17.7
## 8   Female   17.0
## 9     Male   20.0
## 10    Male   18.5
## 11  Female   17.0
## 12    Male   21.0
## 13  Female   16.0
## 14  Female   19.5
## 15    Male   16.0
## 16  Female   17.5
## 17  Female   18.0
## 18    Male   19.4
## 19    Male   20.5
## 20    Male   21.0
## 21    Male   21.5
## 22    Male   20.1
## 23    Male   18.5
## 24    Male   21.5
## 25  Female   17.0
## 26    Male   18.5
## 27    Male   21.0
## 28    Male   20.8
## 29    Male   17.8
## 30    Male   19.5
## 31  Female   18.5
## 32    Male   18.8
## 33  Female   17.1
## 34    Male   20.1
## 35    Male   18.0
## 36    Male   22.2
## 37  Female   16.0
## 38    Male   19.4
## 39    Male   22.0
## 40    Male   19.0
## 41  Female   17.5
## 42  Female   17.8
## 43    Male     NA
## 44  Female   20.1
## 45  Female   13.0
## 46    Male   17.0
## 47    Male   23.2
## 48    Male   22.5
## 49  Female   18.0
## 50  Female   18.0
## 51    Male   22.0
## 52    Male   20.5
## 53    Male   17.0
## 54    Male   20.5
## 55    Male   22.5
## 56    Male   18.5
```

```
## 57  Female   15.5
## 58    Male   19.5
## 59    Male   19.5
## 60    Male   20.6
## 61    Male   22.8
## 62  Female   18.5
## 63  Female   19.6
## 64  Female   18.7
## 65  Female   17.3
## 66    Male   19.5
## 67  Female   19.0
## 68  Female   18.5
## 69    Male   19.0
## 70    Male   21.0
## 71  Female   18.0
## 72    Male   19.4
## 73  Female   17.0
## 74  Female   16.5
## 75  Female   15.6
## 76  Female   17.5
## 77  Female   17.0
## 78  Female   18.6
## 79  Female   18.3
## 80    Male   20.0
## 81    Male   19.5
## 82    Male   19.2
## 83  Female   17.5
## 84  Female   17.0
## 85    Male   23.0
## 86  Female   17.7
## 87  Female   18.2
## 88  Female   18.3
## 89    Male   18.0
## 90  Female   18.0
## 91    Male   20.5
## 92  Female   17.5
## 93  Female   18.2
## 94  Female   18.2
## 95    Male   21.3
## 96  Female   19.0
## 97    Male   20.0
## 98  Female   17.5
## 99    Male   19.5
## 100 Female   19.4
## 101   Male   21.9
## 102   Male   18.9
## 103 Female   16.0
## 104 Female   17.5
## 105 Female   17.5
## 106 Female   19.5
## 107 Female   16.2
## 108 Female   17.0
## 109   Male   17.5
## 110   Male   19.7
```

```
## 111 Female   18.5
## 112   Male   19.2
## 113 Female   17.2
## 114   Male   20.5
## 115 Female   16.0
## 116 Female   16.9
## 117 Female   17.0
## 118   Male   23.0
## 119 Female   18.5
## 120   Male   21.0
## 121   Male   20.0
## 122   Male   22.5
## 123 Female   18.5
## 124   Male   19.8
## 125   Male   18.5
## 126   Male   19.3
## 127 Female   16.0
## 128   Male   18.8
## 129 Female   17.5
## 130 Female   16.4
## 131   Male   22.0
## 132   Male   19.0
## 133 Female   18.9
## 134 Female   15.4
## 135   Male   17.9
## 136   Male   23.1
## 137   <NA>   19.8
## 138   Male   22.0
## 139   Male   20.0
## 140 Female   19.5
## 141 Female   18.0
## 142 Female   18.3
## 143 Female   19.0
## 144   Male   21.4
## 145 Female   20.0
## 146   Male   18.5
## 147   Male   22.5
## 148   Male   19.5
## 149 Female   18.0
## 150 Female   18.0
## 151   Male   21.8
## 152 Female   13.0
## 153 Female   16.3
## 154   Male   21.5
## 155   Male   18.9
## 156   Male   20.5
## 157   Male   14.0
## 158 Female   18.9
## 159   Male   20.0
## 160   Male   18.5
## 161 Female   17.5
## 162   Male   18.1
## 163   Male   20.2
## 164 Female   16.5
```

```
## 165   Male   19.1
## 166 Female   17.6
## 167 Female   19.5
## 168 Female   16.5
## 169   Male   19.0
## 170   Male   19.0
## 171 Female   16.5
## 172   Male   20.5
## 173 Female   15.5
## 174 Female   18.0
## 175 Female   17.5
## 176 Female   19.0
## 177   Male   20.5
## 178 Female   16.7
## 179 Female   20.5
## 180 Female   17.0
## 181   Male   19.0
## 182 Female   14.0
## 183 Female   17.5
## 184   Male   18.5
## 185   Male   18.0
## 186   Male   20.5
## 187 Female   17.0
## 188   Male   18.5
## 189   Male   18.0
## 190   Male   18.5
## 191   Male   20.0
## 192   Male   22.0
## 193   Male   17.9
## 194 Female   17.6
## 195 Female   16.7
## 196 Female   17.0
## 197 Female   15.0
## 198   Male   16.0
## 199 Female   19.1
## 200 Female   17.5
## 201 Female   16.2
## 202   Male   21.0
## 203 Female   18.8
## 204 Female   18.5
## 205   Male   17.0
## 206 Female   17.5
## 207 Female   17.5
## 208   Male   17.5
## 209   Male   17.5
## 210 Female   20.8
## 211 Female   18.6
## 212 Female   17.5
## 213   Male   18.0
## 214   Male   17.0
## 215 Female   18.0
## 216   Male   19.5
## 217 Female   16.3
## 218   Male   18.2
```

```
## 219 Female   17.0
## 220   Male   23.2
## 221   Male   23.2
## 222 Female   15.9
## 223 Female   17.5
## 224 Female   17.5
## 225 Female   17.6
## 226 Female   17.5
## 227 Female   18.8
## 228   Male   20.0
## 229 Female   18.6
## 230   Male   18.6
## 231 Female   18.8
## 232   Male   18.0
## 233 Female   18.0
## 234 Female   18.5
## 235 Female   17.5
## 236   Male   21.0
## 237 Female   17.6
```

```
mydata[ ,c("Sex", "Fold")] # by name like a matrix
```

```
##         Sex    Fold
## 1   Female  R on L
## 2     Male  R on L
## 3     Male  L on R
## 4     Male  R on L
## 5     Male Neither
## 6   Female  L on R
## 7     Male  L on R
## 8   Female  R on L
## 9     Male  R on L
## 10    Male  R on L
## 11  Female  L on R
## 12    Male  R on L
## 13  Female  L on R
## 14  Female  L on R
## 15    Male  R on L
## 16  Female  R on L
## 17  Female  L on R
## 18    Male  R on L
## 19    Male  L on R
## 20    Male  R on L
## 21    Male  R on L
## 22    Male  L on R
## 23    Male  L on R
## 24    Male  R on L
## 25  Female  R on L
## 26    Male Neither
## 27    Male  R on L
## 28    Male  R on L
## 29    Male  L on R
## 30    Male  L on R
## 31  Female  R on L
## 32    Male  L on R
```

```
## 33  Female  R on L
## 34    Male  R on L
## 35    Male  L on R
## 36    Male  L on R
## 37  Female  L on R
## 38    Male  R on L
## 39    Male  R on L
## 40    Male  R on L
## 41  Female  L on R
## 42  Female  R on L
## 43    Male  R on L
## 44  Female  L on R
## 45  Female  L on R
## 46    Male  R on L
## 47    Male  L on R
## 48    Male  R on L
## 49  Female  R on L
## 50  Female  R on L
## 51    Male  R on L
## 52    Male  L on R
## 53    Male  L on R
## 54    Male  L on R
## 55    Male  R on L
## 56    Male  L on R
## 57  Female  R on L
## 58    Male  R on L
## 59    Male  L on R
## 60    Male  L on R
## 61    Male  R on L
## 62  Female  R on L
## 63  Female  L on R
## 64  Female  L on R
## 65  Female  L on R
## 66    Male Neither
## 67  Female  L on R
## 68  Female  R on L
## 69    Male  L on R
## 70    Male  L on R
## 71  Female  L on R
## 72    Male  R on L
## 73  Female  R on L
## 74  Female  L on R
## 75  Female  R on L
## 76  Female Neither
## 77  Female  L on R
## 78  Female  L on R
## 79  Female  R on L
## 80    Male  L on R
## 81    Male  R on L
## 82    Male  R on L
## 83  Female  R on L
## 84  Female  R on L
## 85    Male  L on R
## 86  Female  R on L
```

```
## 87  Female  L on R
## 88  Female  R on L
## 89    Male Neither
## 90  Female  R on L
## 91    Male  R on L
## 92  Female Neither
## 93  Female  L on R
## 94  Female  R on L
## 95    Male  R on L
## 96  Female  L on R
## 97    Male  R on L
## 98  Female  R on L
## 99    Male Neither
## 100 Female  R on L
## 101   Male  R on L
## 102   Male  L on R
## 103 Female Neither
## 104 Female  R on L
## 105 Female  R on L
## 106 Female  R on L
## 107 Female  R on L
## 108 Female  R on L
## 109   Male  L on R
## 110   Male  R on L
## 111 Female  R on L
## 112   Male  L on R
## 113 Female  R on L
## 114   Male  R on L
## 115 Female  L on R
## 116 Female  L on R
## 117 Female  R on L
## 118   Male  L on R
## 119 Female  L on R
## 120   Male  L on R
## 121   Male  R on L
## 122   Male  L on R
## 123 Female  R on L
## 124   Male  L on R
## 125   Male  L on R
## 126   Male  R on L
## 127 Female  R on L
## 128   Male  L on R
## 129 Female  R on L
## 130 Female  L on R
## 131   Male  R on L
## 132   Male  L on R
## 133 Female  R on L
## 134 Female  L on R
## 135   Male  R on L
## 136   Male  L on R
## 137   <NA>  L on R
## 138   Male  L on R
## 139   Male  L on R
## 140 Female  L on R
```

```
## 141 Female  R on L
## 142 Female  R on L
## 143 Female  R on L
## 144   Male  L on R
## 145 Female  R on L
## 146   Male  R on L
## 147   Male  L on R
## 148   Male  R on L
## 149 Female  L on R
## 150 Female  R on L
## 151   Male  R on L
## 152 Female  L on R
## 153 Female  L on R
## 154   Male  R on L
## 155   Male  L on R
## 156   Male  R on L
## 157   Male  L on R
## 158 Female  L on R
## 159   Male  R on L
## 160   Male  L on R
## 161 Female  R on L
## 162   Male Neither
## 163   Male  L on R
## 164 Female  R on L
## 165   Male Neither
## 166 Female  R on L
## 167 Female  R on L
## 168 Female  L on R
## 169   Male  L on R
## 170   Male  R on L
## 171 Female  L on R
## 172   Male  L on R
## 173 Female Neither
## 174 Female  R on L
## 175 Female  R on L
## 176 Female  L on R
## 177   Male Neither
## 178 Female  L on R
## 179 Female  R on L
## 180 Female  R on L
## 181   Male  R on L
## 182 Female  R on L
## 183 Female  L on R
## 184   Male  L on R
## 185   Male Neither
## 186   Male  R on L
## 187 Female  L on R
## 188   Male  R on L
## 189   Male  R on L
## 190   Male Neither
## 191   Male  R on L
## 192   Male  L on R
## 193   Male  R on L
## 194 Female  L on R
```

```
## 195 Female Neither
## 196 Female  L on R
## 197 Female  R on L
## 198   Male Neither
## 199 Female  R on L
## 200 Female  R on L
## 201 Female  R on L
## 202   Male  L on R
## 203 Female  R on L
## 204 Female Neither
## 205   Male  R on L
## 206 Female  R on L
## 207 Female  L on R
## 208   Male  R on L
## 209   Male  L on R
## 210 Female  R on L
## 211 Female  L on R
## 212 Female  R on L
## 213   Male  R on L
## 214   Male  R on L
## 215 Female  L on R
## 216   Male Neither
## 217 Female  L on R
## 218   Male  R on L
## 219 Female  L on R
## 220   Male  L on R
## 221   Male  L on R
## 222 Female  R on L
## 223 Female  R on L
## 224 Female  L on R
## 225 Female  L on R
## 226 Female  R on L
## 227 Female  R on L
## 228   Male  L on R
## 229 Female  L on R
## 230   Male  L on R
## 231 Female  R on L
## 232   Male  R on L
## 233 Female  L on R
## 234 Female  L on R
## 235 Female  R on L
## 236   Male  R on L
## 237 Female  R on L
```

```
mydata[c("Sex", "Fold")] #
```

```
##        Sex    Fold
## 1   Female  R on L
## 2     Male  R on L
## 3     Male  L on R
## 4     Male  R on L
## 5     Male Neither
## 6   Female  L on R
## 7     Male  L on R
## 8   Female  R on L
```

```
## 9      Male  R on L
## 10     Male  R on L
## 11   Female  L on R
## 12     Male  R on L
## 13   Female  L on R
## 14   Female  L on R
## 15     Male  R on L
## 16   Female  R on L
## 17   Female  L on R
## 18     Male  R on L
## 19     Male  L on R
## 20     Male  R on L
## 21     Male  R on L
## 22     Male  L on R
## 23     Male  L on R
## 24     Male  R on L
## 25   Female  R on L
## 26     Male Neither
## 27     Male  R on L
## 28     Male  R on L
## 29     Male  L on R
## 30     Male  L on R
## 31   Female  R on L
## 32     Male  L on R
## 33   Female  R on L
## 34     Male  R on L
## 35     Male  L on R
## 36     Male  L on R
## 37   Female  L on R
## 38     Male  R on L
## 39     Male  R on L
## 40     Male  R on L
## 41   Female  L on R
## 42   Female  R on L
## 43     Male  R on L
## 44   Female  L on R
## 45   Female  L on R
## 46     Male  R on L
## 47     Male  L on R
## 48     Male  R on L
## 49   Female  R on L
## 50   Female  R on L
## 51     Male  R on L
## 52     Male  L on R
## 53     Male  L on R
## 54     Male  L on R
## 55     Male  R on L
## 56     Male  L on R
## 57   Female  R on L
## 58     Male  R on L
## 59     Male  L on R
## 60     Male  L on R
## 61     Male  R on L
## 62   Female  R on L
```

```
## 63  Female  L on R
## 64  Female  L on R
## 65  Female  L on R
## 66    Male Neither
## 67  Female  L on R
## 68  Female  R on L
## 69    Male  L on R
## 70    Male  L on R
## 71  Female  L on R
## 72    Male  R on L
## 73  Female  R on L
## 74  Female  L on R
## 75  Female  R on L
## 76  Female Neither
## 77  Female  L on R
## 78  Female  L on R
## 79  Female  R on L
## 80    Male  L on R
## 81    Male  R on L
## 82    Male  R on L
## 83  Female  R on L
## 84  Female  R on L
## 85    Male  L on R
## 86  Female  R on L
## 87  Female  L on R
## 88  Female  R on L
## 89    Male Neither
## 90  Female  R on L
## 91    Male  R on L
## 92  Female Neither
## 93  Female  L on R
## 94  Female  R on L
## 95    Male  R on L
## 96  Female  L on R
## 97    Male  R on L
## 98  Female  R on L
## 99    Male Neither
## 100 Female  R on L
## 101   Male  R on L
## 102   Male  L on R
## 103 Female Neither
## 104 Female  R on L
## 105 Female  R on L
## 106 Female  R on L
## 107 Female  R on L
## 108 Female  R on L
## 109   Male  L on R
## 110   Male  R on L
## 111 Female  R on L
## 112   Male  L on R
## 113 Female  R on L
## 114   Male  R on L
## 115 Female  L on R
## 116 Female  L on R
```

```
## 117 Female  R on L
## 118   Male  L on R
## 119 Female  L on R
## 120   Male  L on R
## 121   Male  R on L
## 122   Male  L on R
## 123 Female  R on L
## 124   Male  L on R
## 125   Male  L on R
## 126   Male  R on L
## 127 Female  R on L
## 128   Male  L on R
## 129 Female  R on L
## 130 Female  L on R
## 131   Male  R on L
## 132   Male  L on R
## 133 Female  R on L
## 134 Female  L on R
## 135   Male  R on L
## 136   Male  L on R
## 137   <NA>  L on R
## 138   Male  L on R
## 139   Male  L on R
## 140 Female  L on R
## 141 Female  R on L
## 142 Female  R on L
## 143 Female  R on L
## 144   Male  L on R
## 145 Female  R on L
## 146   Male  R on L
## 147   Male  L on R
## 148   Male  R on L
## 149 Female  L on R
## 150 Female  R on L
## 151   Male  R on L
## 152 Female  L on R
## 153 Female  L on R
## 154   Male  R on L
## 155   Male  L on R
## 156   Male  R on L
## 157   Male  L on R
## 158 Female  L on R
## 159   Male  R on L
## 160   Male  L on R
## 161 Female  R on L
## 162   Male Neither
## 163   Male  L on R
## 164 Female  R on L
## 165   Male Neither
## 166 Female  R on L
## 167 Female  R on L
## 168 Female  L on R
## 169   Male  L on R
## 170   Male  R on L
```

```
## 171 Female  L on R
## 172   Male  L on R
## 173 Female Neither
## 174 Female  R on L
## 175 Female  R on L
## 176 Female  L on R
## 177   Male Neither
## 178 Female  L on R
## 179 Female  R on L
## 180 Female  R on L
## 181   Male  R on L
## 182 Female  R on L
## 183 Female  L on R
## 184   Male  L on R
## 185   Male Neither
## 186   Male  R on L
## 187 Female  L on R
## 188   Male  R on L
## 189   Male  R on L
## 190   Male Neither
## 191   Male  R on L
## 192   Male  L on R
## 193   Male  R on L
## 194 Female  L on R
## 195 Female Neither
## 196 Female  L on R
## 197 Female  R on L
## 198   Male Neither
## 199 Female  R on L
## 200 Female  R on L
## 201 Female  R on L
## 202   Male  L on R
## 203 Female  R on L
## 204 Female Neither
## 205   Male  R on L
## 206 Female  R on L
## 207 Female  L on R
## 208   Male  R on L
## 209   Male  L on R
## 210 Female  R on L
## 211 Female  L on R
## 212 Female  R on L
## 213   Male  R on L
## 214   Male  R on L
## 215 Female  L on R
## 216   Male Neither
## 217 Female  L on R
## 218   Male  R on L
## 219 Female  L on R
## 220   Male  L on R
## 221   Male  L on R
## 222 Female  R on L
## 223 Female  R on L
## 224 Female  L on R
```

```
## 225 Female  L on R
## 226 Female  R on L
## 227 Female  R on L
## 228   Male  L on R
## 229 Female  L on R
## 230   Male  L on R
## 231 Female  R on L
## 232   Male  R on L
## 233 Female  L on R
## 234 Female  L on R
## 235 Female  R on L
## 236   Male  R on L
## 237 Female  R on L
```

```
mydata$Sex # by name to get a single variable
```

```
##   [1] Female Male   Male   Male   Male   Female Male   Female Male   Male
##  [11] Female Male   Female Female Male   Female Female Male   Male   Male
##  [21] Male   Male   Male   Male   Female Male   Male   Male   Male   Male
##  [31] Female Male   Female Male   Male   Male   Female Male   Male   Male
##  [41] Female Female Male   Female Female Male   Male   Male   Female Female
##  [51] Male   Male   Male   Male   Male   Male   Female Male   Male   Male
##  [61] Male   Female Female Female Female Male   Female Female Male   Male
##  [71] Female Male   Female Female Female Female Female Female Female Male
##  [81] Male   Male   Female Female Male   Female Female Female Male   Female
##  [91] Male   Female Female Female Male   Female Male   Female Male   Female
## [101] Male   Male   Female Female Female Female Female Female Male   Male
## [111] Female Male   Female Male   Female Female Female Male   Female Male
## [121] Male   Male   Female Male   Male   Male   Female Male   Female Female
## [131] Male   Male   Female Female Male   Male   <NA>   Male   Male   Female
## [141] Female Female Female Male   Female Male   Male   Male   Female Female
## [151] Male   Female Female Male   Male   Male   Male   Female Male   Male
## [161] Female Male   Male   Female Male   Female Female Female Male   Male
## [171] Female Male   Female Female Female Female Male   Female Female Female
## [181] Male   Female Female Male   Male   Male   Female Male   Male   Male
## [191] Male   Male   Male   Female Female Female Female Male   Female Female
## [201] Female Male   Female Female Male   Female Female Male   Male   Female
## [211] Female Female Male   Male   Female Male   Female Male   Female Male
## [221] Male   Female Female Female Female Female Female Male   Female Male
## [231] Female Male   Female Female Female Male   Female
## Levels: Female Male
```

```
##################################################
# Names
names(mydata) # get variable names
```

```
##  [1] "Sex"    "Wr.Hnd" "NW.Hnd" "W.Hnd"  "Fold"   "Pulse"  "Clap"
##  [8] "Exer"   "Smoke"  "Height" "M.I"    "Age"
```

```
colnames(mydata) # but this also works
```

```
##  [1] "Sex"    "Wr.Hnd" "NW.Hnd" "W.Hnd"  "Fold"   "Pulse"  "Clap"
##  [8] "Exer"   "Smoke"  "Height" "M.I"    "Age"
```

```
rownames(mydata) # rows can also have names
```

```
##   [1] "1"   "2"   "3"   "4"   "5"   "6"   "7"   "8"   "9"   "10"  "11"
```

```
##  [12] "12"  "13"  "14"  "15"  "16"  "17"  "18"  "19"  "20"  "21"  "22"
##  [23] "23"  "24"  "25"  "26"  "27"  "28"  "29"  "30"  "31"  "32"  "33"
##  [34] "34"  "35"  "36"  "37"  "38"  "39"  "40"  "41"  "42"  "43"  "44"
##  [45] "45"  "46"  "47"  "48"  "49"  "50"  "51"  "52"  "53"  "54"  "55"
##  [56] "56"  "57"  "58"  "59"  "60"  "61"  "62"  "63"  "64"  "65"  "66"
##  [67] "67"  "68"  "69"  "70"  "71"  "72"  "73"  "74"  "75"  "76"  "77"
##  [78] "78"  "79"  "80"  "81"  "82"  "83"  "84"  "85"  "86"  "87"  "88"
##  [89] "89"  "90"  "91"  "92"  "93"  "94"  "95"  "96"  "97"  "98"  "99"
## [100] "100" "101" "102" "103" "104" "105" "106" "107" "108" "109" "110"
## [111] "111" "112" "113" "114" "115" "116" "117" "118" "119" "120" "121"
## [122] "122" "123" "124" "125" "126" "127" "128" "129" "130" "131" "132"
## [133] "133" "134" "135" "136" "137" "138" "139" "140" "141" "142" "143"
## [144] "144" "145" "146" "147" "148" "149" "150" "151" "152" "153" "154"
## [155] "155" "156" "157" "158" "159" "160" "161" "162" "163" "164" "165"
## [166] "166" "167" "168" "169" "170" "171" "172" "173" "174" "175" "176"
## [177] "177" "178" "179" "180" "181" "182" "183" "184" "185" "186" "187"
## [188] "188" "189" "190" "191" "192" "193" "194" "195" "196" "197" "198"
## [199] "199" "200" "201" "202" "203" "204" "205" "206" "207" "208" "209"
## [210] "210" "211" "212" "213" "214" "215" "216" "217" "218" "219" "220"
## [221] "221" "222" "223" "224" "225" "226" "227" "228" "229" "230" "231"
## [232] "232" "233" "234" "235" "236" "237"
```

```r
# Tip: Avoid row names.
# Add another variable to the data.frame to store this information.

# Examine first few rows
head(mydata) # first 6 rows
```

```
##      Sex Wr.Hnd NW.Hnd W.Hnd    Fold Pulse    Clap Exer Smoke Height
## 1 Female   18.5   18.0 Right  R on L    92    Left Some Never 173.00
## 2   Male   19.5   20.5  Left  R on L   104    Left None Regul 177.80
## 3   Male   18.0   13.3 Right  L on R    87 Neither None Occas     NA
## 4   Male   18.8   18.9 Right  R on L    NA Neither None Never 160.00
## 5   Male   20.0   20.0 Right Neither    35   Right Some Never 165.00
## 6 Female   18.0   17.7 Right  L on R    64   Right Some Never 172.72
##        M.I    Age
## 1   Metric 18.250
## 2 Imperial 17.583
## 3     <NA> 16.917
## 4   Metric 20.333
## 5   Metric 23.667
## 6 Imperial 21.000
```

```r
head(mydata, n = 10) # first 7 rows
```

```
##       Sex Wr.Hnd NW.Hnd W.Hnd    Fold Pulse    Clap Exer Smoke Height
## 1  Female   18.5   18.0 Right  R on L    92    Left Some Never 173.00
## 2    Male   19.5   20.5  Left  R on L   104    Left None Regul 177.80
## 3    Male   18.0   13.3 Right  L on R    87 Neither None Occas     NA
## 4    Male   18.8   18.9 Right  R on L    NA Neither None Never 160.00
## 5    Male   20.0   20.0 Right Neither    35   Right Some Never 165.00
## 6  Female   18.0   17.7 Right  L on R    64   Right Some Never 172.72
## 7    Male   17.7   17.7 Right  L on R    83   Right Freq Never 182.88
## 8  Female   17.0   17.3 Right  R on L    74   Right Freq Never 157.00
## 9    Male   20.0   19.5 Right  R on L    72   Right Some Never 175.00
```

```
## 10   Male   18.5   18.5 Right  R on L    90   Right Some Never 167.00
##         M.I    Age
## 1    Metric 18.250
## 2  Imperial 17.583
## 3      <NA> 16.917
## 4    Metric 20.333
## 5    Metric 23.667
## 6  Imperial 21.000
## 7  Imperial 18.833
## 8    Metric 35.833
## 9    Metric 19.000
## 10   Metric 22.333
```

```r
tail(mydata) # last few rows
```

```
##        Sex Wr.Hnd NW.Hnd W.Hnd   Fold Pulse  Clap Exer Smoke Height
## 232   Male   18.0   16.0 Right R on L    NA Right Some Never 180.34
## 233 Female   18.0   18.0 Right L on R    85 Right Some Never 165.10
## 234 Female   18.5   18.0 Right L on R    88 Right Some Never 160.00
## 235 Female   17.5   16.5 Right R on L    NA Right Some Never 170.00
## 236   Male   21.0   21.5 Right R on L    90 Right Some Never 183.00
## 237 Female   17.6   17.3 Right R on L    85 Right Freq Never 168.50
##          M.I    Age
## 232 Imperial 20.750
## 233 Imperial 17.667
## 234   Metric 16.917
## 235   Metric 18.583
## 236   Metric 17.167
## 237   Metric 17.750
```

```r
# View(mydata) # Rstudio function to open data in viewer
# or click on the icon in the Environment pane

# How many rows and columns?
dim(mydata) # rows and column counts
```

```
## [1] 237  12
```

```r
nrow(mydata) # row count
```

```
## [1] 237
```

```r
ncol(mydata) # column count
```

```
## [1] 12
```

```r
# Examine structure
str(mydata)
```

```
## 'data.frame':    237 obs. of  12 variables:
##  $ Sex   : Factor w/ 2 levels "Female","Male": 1 2 2 2 2 1 2 1 2 2 ...
##  $ Wr.Hnd: num  18.5 19.5 18 18.8 20 18 17.7 17 20 18.5 ...
##  $ NW.Hnd: num  18 20.5 13.3 18.9 20 17.7 17.7 17.3 19.5 18.5 ...
##  $ W.Hnd : Factor w/ 2 levels "Left","Right": 2 1 2 2 2 2 2 2 2 2 ...
##  $ Fold  : Factor w/ 3 levels "L on R","Neither",..: 3 3 1 3 2 1 1 3 3 3 ...
##  $ Pulse : int  92 104 87 NA 35 64 83 74 72 90 ...
##  $ Clap  : Factor w/ 3 levels "Left","Neither",..: 1 1 2 2 3 3 3 3 3 3 ...
##  $ Exer  : Factor w/ 3 levels "Freq","None",..: 3 2 2 2 3 3 1 1 3 3 ...
```

```
##  $ Smoke : Factor w/ 4 levels "Heavy","Never",..: 2 4 3 2 2 2 2 2 2 2 ...
##  $ Height: num  173 178 NA 160 165 ...
##  $ M.I   : Factor w/ 2 levels "Imperial","Metric": 2 1 NA 2 2 1 1 2 2 2 ...
##  $ Age   : num  18.2 17.6 16.9 20.3 23.7 ...
```

Getting help

```r
# Use question mark (i.e., ?) followed by command name
# to lookup specific command
?mean
help(mean) # or use help function

# to look up package
help(package = "MASS")

# Press F1 in RStudio on the command name
# mean

# Use double question mark to do a full-text search on R help
??"factor analysis"

# Search google
# e.g., how to get the mean of a vector using r

# Ask question on Stackoverflow with the R tag
# http://stackoverflow.com/questions/tagged/r
```

Exercise 1

```r
# 1. Working with vectors
# 1.1 Create a variable called x with 10 values drawn from a
#    normal distribution (see rnorm)

# 1.2 Use the sum and > operator to work out how many values in x
#    are larger than 1


# 3. Using the survey dataset in the MASS package
library(MASS)
data(survey)
# 3.1 Look up the help file on MASS

# 3.2 How many observations are there?

# 3.3 Show the first 10 rows of the cats data.frame

# 3.4 Show the structure of cats using the str function

# 3.5 Extract the female cats and assign to variable fcats
```

```r
# 3.6 How many rows is in fcats?
```

Answers 1

```r
# 1. Working with vectors
# 1.1 Create a variable called x with 10 values drawn from a
#    normal distribution (see rnorm)
x <- rnorm(10)

# 1.2 Use the sum and > operator to work out how many values in x
#    are larger than 1
sum(x > 1)
```

```
## [1] 0
```

```r
# 3. Using the cats dataset in the MASS package
library(MASS)
data(survey)
# 3.1 Look up the help file on survey
?survey

# 3.2 How many observations are there?
nrow(survey)
```

```
## [1] 237
```

```r
# 3.3 Show the first 10 rows of the survey data.frame
head(survey, 10)
```

```
##       Sex Wr.Hnd NW.Hnd W.Hnd    Fold Pulse    Clap Exer Smoke Height
## 1  Female   18.5   18.0 Right  R on L    92    Left Some Never 173.00
## 2    Male   19.5   20.5  Left  R on L   104    Left None Regul 177.80
## 3    Male   18.0   13.3 Right  L on R    87 Neither None Occas     NA
## 4    Male   18.8   18.9 Right  R on L    NA Neither None Never 160.00
## 5    Male   20.0   20.0 Right Neither    35   Right Some Never 165.00
## 6  Female   18.0   17.7 Right  L on R    64   Right Some Never 172.72
## 7    Male   17.7   17.7 Right  L on R    83   Right Freq Never 182.88
## 8  Female   17.0   17.3 Right  R on L    74   Right Freq Never 157.00
## 9    Male   20.0   19.5 Right  R on L    72   Right Some Never 175.00
## 10   Male   18.5   18.5 Right  R on L    90   Right Some Never 167.00
##         M.I    Age
## 1    Metric 18.250
## 2  Imperial 17.583
## 3      <NA> 16.917
## 4    Metric 20.333
## 5    Metric 23.667
## 6  Imperial 21.000
## 7  Imperial 18.833
## 8    Metric 35.833
## 9    Metric 19.000
## 10   Metric 22.333
```

```r
# 3.4 Show the structure of survey using the str function
str(survey)
```

```
## 'data.frame':    237 obs. of  12 variables:
##  $ Sex   : Factor w/ 2 levels "Female","Male": 1 2 2 2 2 1 2 1 2 2 ...
##  $ Wr.Hnd: num  18.5 19.5 18 18.8 20 18 17.7 17 20 18.5 ...
##  $ NW.Hnd: num  18 20.5 13.3 18.9 20 17.7 17.7 17.3 19.5 18.5 ...
##  $ W.Hnd : Factor w/ 2 levels "Left","Right": 2 1 2 2 2 2 2 2 2 2 ...
##  $ Fold  : Factor w/ 3 levels "L on R","Neither",..: 3 3 1 3 2 1 1 3 3 3 ...
##  $ Pulse : int  92 104 87 NA 35 64 83 74 72 90 ...
##  $ Clap  : Factor w/ 3 levels "Left","Neither",..: 1 1 2 2 3 3 3 3 3 3 ...
##  $ Exer  : Factor w/ 3 levels "Freq","None",..: 3 2 2 2 3 3 1 1 3 3 ...
##  $ Smoke : Factor w/ 4 levels "Heavy","Never",..: 2 4 3 2 2 2 2 2 2 2 ...
##  $ Height: num  173 178 NA 160 165 ...
##  $ M.I   : Factor w/ 2 levels "Imperial","Metric": 2 1 NA 2 2 1 1 2 2 2 ...
##  $ Age   : num  18.2 17.6 16.9 20.3 23.7 ...
```

```r
# 3.5 Extract the female survey and assign to variable fsurvey
fsurvey <- survey[ survey$Sex == "F", ]

# 3.6 How many rows is in fsurvey?
nrow(fsurvey)
```

```
## [1] 1
```

Packages

```r
# R has many additional packages
# To use a package it needs to be installed.
# You only need to install a package once.
# To use a package, you need to load the package each time
# you use R.


#############################################################
# Installing an R package
# Option 1. Use the install.packages function.
# install.packages("psych")

# Option 2. Use the package tab in R Studio
# Click install and enter package details



#############################################################
# Loading an installed package
# Option 1. Use the library function
library(psych) # I.e., put this at the start of your script

# Other options
# 2. We may talk about ProjectTemplate later
#
```

```r
# 3. Put it in your R startup file
#    (not recommended as it reduces reproducibility)

##################################################################
# Common errors
# Not having a package installed is a common error
# If you try to load a package that is not installed.
# e.g.,
# library(foo)
# You will get an error
# Error in library(foo) : there is no package called ‘foo’
# This means either
# 1. You mistyped the name of the package, or
# 2. You need to install the pakcage
#     install.packages("foo")
# Note foo is just an example. There is no package called foo.

# Trying to use a function from a package that is not loaded is a common error

# E.g., there is a function you want to use
detach(package:psych) # used for example to ensure psych is not attached
# say we wanted to use the fisherz function from the psych package
# but the psych package is not loaded
fisherz(.3)
```

```
## Error in fisherz(0.3): could not find function "fisherz"
```

```r
# We get the error:
#   "Error: object 'fisherz' not found"
# Thus we need to run
library(psych)
```

```
## Warning: package 'psych' was built under R version 3.5.2

##
## Attaching package: 'psych'

## The following object is masked from 'package:boot':
##
##     logit

## The following object is masked from 'package:lavaan':
##
##     cor2cov

## The following object is masked from 'package:car':
##
##     logit

## The following object is masked from 'package:Hmisc':
##
##     describe

## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```r
fisherz(.3)
```

```
## [1] 0.3095196
```

```r
###############################################################
# Packages contain additional functions.
# Once the package is loaded, functions are added to the workspace
# list workspace
search()
```

```
##  [1] ".GlobalEnv"        "package:psych"     "package:boot"
##  [4] "package:metafor"   "package:lavaan"    "package:lme4"
##  [7] "package:Matrix"    "package:dplyr"     "package:MASS"
## [10] "package:AER"       "package:sandwich"  "package:lmtest"
## [13] "package:zoo"       "package:car"       "package:carData"
## [16] "package:Hmisc"     "package:ggplot2"   "package:Formula"
## [19] "package:survival"  "package:lattice"   "package:stats"
## [22] "package:graphics"  "package:grDevices" "package:utils"
## [25] "package:datasets"  "package:methods"   "Autoloads"
## [28] "package:base"
```

```r
# To make it clear that a function comes from a particular package
# or to overcome the issue where two packages have functions with the same names
# use double colon (i.e., package::function).
# RStudio also permits auto-completion of function names.
psych::alpha # alpha is a funtion in the psych package
```

```
## function (x, keys = NULL, cumulative = FALSE, title = NULL, max = 10,
##     na.rm = TRUE, check.keys = FALSE, n.iter = 1, delete = TRUE,
##     use = "pairwise", warnings = TRUE, n.obs = NULL, impute = NULL)
## {
##     alpha.1 <- function(C, R) {
##         n <- dim(C)[2]
##         alpha.raw <- (1 - tr(C)/sum(C)) * (n/(n - 1))
##         sumR <- sum(R)
##         alpha.std <- (1 - n/sumR) * (n/(n - 1))
##         smc.R <- smc(R)
##         G6 <- (1 - (n - sum(smc.R))/sumR)
##         av.r <- (sumR - n)/(n * (n - 1))
##         R.adj <- R[lower.tri(R)]
##         var.r <- var(R.adj, na.rm = TRUE)
##         med.r <- median(R.adj, na.rm = TRUE)
##         mod1 <- matrix(av.r, n, n)
##         Res1 <- R - mod1
##         GF1 = 1 - sum(Res1^2)/sum(R^2)
##         Rd <- R - diag(R)
##         diag(Res1) <- 0
##         GF1.off <- 1 - sum(Res1^2)/sum(Rd^2)
##         sn <- n * av.r/(1 - av.r)
##         Q = (2 * n^2/((n - 1)^2 * (sum(C)^3))) * (sum(C) * (tr(C %*%
##             C) + (tr(C))^2) - 2 * (tr(C) * sum(C %*% C)))
##         result <- list(raw = alpha.raw, std = alpha.std, G6 = G6,
##             av.r = av.r, sn = sn, Q = Q, GF1, GF1.off, var.r = var.r,
##             med.r = med.r)
##         return(result)
```

```
##     }
##     cl <- match.call()
##     if (!is.matrix(x) && !is.data.frame(x))
##         stop("Data must either be a data frame or a matrix")
##     if (class(x)[1] != "data.frame")
##         x <- fix.dplyr(x)
##     if (!is.null(keys)) {
##         if (is.list(keys)) {
##             select <- sub("-", "", unlist(keys))
##             x <- x[, select]
##             keys <- make.keys(x, keys)
##         }
##         else {
##             temp <- rep(1, ncol(x))
##             temp[(colnames(x) %in% keys)] <- -1
##             keys <- temp
##         }
##     }
##     nvar <- dim(x)[2]
##     nsub <- dim(x)[1]
##     scores <- NULL
##     response.freq <- NULL
##     raw <- FALSE
##     if (!isCorrelation(x)) {
##         raw <- TRUE
##         if (!is.null(impute)) {
##             if (impute == "median") {
##                 item.impute <- apply(x, 2, median, na.rm = na.rm)
##             }
##             else {
##                 item.impute <- apply(x, 2, mean, na.rm = na.rm)
##             }
##             for (i in 1:nsub) {
##                 for (j in 1:nvar) {
##                   x[i, is.na(x[i, j])] <- item.impute[j]
##                 }
##             }
##         }
##         item.var <- apply(x, 2, sd, na.rm = na.rm)
##         bad <- which((item.var <= 0) | is.na(item.var))
##         if ((length(bad) > 0) && delete) {
##             for (baddy in 1:length(bad)) {
##                 warning("Item = ", colnames(x)[bad][baddy], " had no variance and was deleted")
##             }
##             x <- x[, -bad]
##             nvar <- nvar - length(bad)
##         }
##         response.freq <- response.frequencies(x, max = max)
##         C <- cov(x, use = use)
##     }
##     else {
##         C <- x
##     }
##     if (is.null(colnames(x)))
```

```
##         colnames(x) <- paste0("V", 1:nvar)
##     p1 <- principal(x, scores = FALSE)
##     if (any(p1$loadings < 0)) {
##         if (check.keys) {
##             if (warnings)
##                 warning("Some items were negatively correlated with total scale and were automaticall
##             keys <- 1 - 2 * (p1$loadings < 0)
##         }
##         else {
##             if (is.null(keys) && warnings) {
##                 warning("Some items were negatively correlated with the total scale and probably \nsh
##                 if (warnings)
##                   cat("Some items (", rownames(p1$loadings)[(p1$loadings <
##                     0)], ") were negatively correlated with the total scale and \nprobably should be
##                 keys <- rep(1, nvar)
##             }
##         }
##     }
##     if (is.null(keys)) {
##         keys <- rep(1, nvar)
##         names(keys) <- colnames(x)
##     }
##     else {
##         keys <- as.vector(keys)
##         names(keys) <- colnames(x)
##         if (length(keys) < nvar) {
##             temp <- keys
##             keys <- rep(1, nvar)
##             names(keys) <- colnames(x)
##             keys[temp] <- -1
##         }
##     }
##     key.d <- diag(keys)
##     C <- key.d %*% C %*% key.d
##     signkey <- strtrim(keys, 1)
##     signkey[signkey == "1"] <- ""
##     colnames(x) <- paste(colnames(x), signkey, sep = "")
##     if (raw) {
##         if (any(keys < 0)) {
##             min.item <- min(x, na.rm = na.rm)
##             max.item <- max(x, na.rm = na.rm)
##             adjust <- max.item + min.item
##             flip.these <- which(keys < 0)
##             x[, flip.these] <- adjust - x[, flip.these]
##         }
##         if (cumulative) {
##             total <- rowSums(x, na.rm = na.rm)
##         }
##         else {
##             total <- rowMeans(x, na.rm = na.rm)
##         }
##         mean.t <- mean(total, na.rm = na.rm)
##         sdev <- sd(total, na.rm = na.rm)
##         raw.r <- cor(total, x, use = use)
```

```
##         t.valid <- colSums(!is.na(x))
##     }
##     else {
##         total <- NULL
##         totals <- TRUE
##     }
##     R <- cov2cor(C)
##     drop.item <- vector("list", nvar)
##     alpha.total <- alpha.1(C, R)
##     if (nvar > 2) {
##         for (i in 1:nvar) {
##             drop.item[[i]] <- alpha.1(C[-i, -i, drop = FALSE],
##                 R[-i, -i, drop = FALSE])
##         }
##     }
##     else {
##         drop.item[[1]] <- drop.item[[2]] <- c(rep(R[1, 2], 2),
##             smc(R)[1], R[1, 2], NA, NA, NA, NA)
##     }
##     by.item <- data.frame(matrix(unlist(drop.item), ncol = 10,
##         byrow = TRUE))
##     if (max(nsub, n.obs) > nvar) {
##         by.item[6] <- sqrt(by.item[6]/(max(nsub, n.obs)))
##         by.item <- by.item[-c(7:8)]
##         colnames(by.item) <- c("raw_alpha", "std.alpha", "G6(smc)",
##             "average_r", "S/N", "alpha se", "var.r", "med.r")
##     }
##     else {
##         by.item <- by.item[-c(6:8)]
##         colnames(by.item) <- c("raw_alpha", "std.alpha", "G6(smc)",
##             "average_r", "S/N", "var.r", "med.r")
##     }
##     rownames(by.item) <- colnames(x)
##     Vt <- sum(R)
##     item.r <- colSums(R)/sqrt(Vt)
##     RC <- R
##     diag(RC) <- smc(R)
##     Vtc <- sum(RC)
##     item.rc <- colSums(RC)/sqrt(Vtc)
##     if (nvar > 1) {
##         r.drop <- rep(0, nvar)
##         for (i in 1:nvar) {
##             v.drop <- sum(C[-i, -i, drop = FALSE])
##             c.drop <- sum(C[, i]) - C[i, i]
##             r.drop[i] <- c.drop/sqrt(C[i, i] * v.drop)
##         }
##     }
##     item.means <- colMeans(x, na.rm = na.rm)
##     item.sd <- apply(x, 2, sd, na.rm = na.rm)
##     if (raw) {
##         Unidim <- alpha.total[7]
##         var.r <- alpha.total[[9]]
##         Fit.off <- alpha.total[8]
##         ase = sqrt(alpha.total$Q/nsub)
```

```
##         alpha.total <- data.frame(alpha.total[1:5], ase = ase,
##             mean = mean.t, sd = sdev, med.r = alpha.total[10])
##         colnames(alpha.total) <- c("raw_alpha", "std.alpha",
##             "G6(smc)", "average_r", "S/N", "ase", "mean", "sd",
##             "median_r")
##         rownames(alpha.total) <- ""
##         stats <- data.frame(n = t.valid, raw.r = t(raw.r), std.r = item.r,
##             r.cor = item.rc, r.drop = r.drop, mean = item.means,
##             sd = item.sd)
##     }
##     else {
##         if (is.null(n.obs)) {
##             Unidim <- alpha.total[7]
##             Fit.off <- alpha.total[8]
##             var.r <- alpha.total[9]
##             med.r <- alpha.total[10]
##             alpha.total <- data.frame(alpha.total[c(1:5, 10)])
##             colnames(alpha.total) <- c("raw_alpha", "std.alpha",
##                 "G6(smc)", "average_r", "S/N", "median_r")
##         }
##         else {
##             Unidim <- alpha.total[7]
##             Fit.off <- alpha.total[8]
##             var.r <- alpha.total[9]
##             alpha.total <- data.frame(alpha.total[1:5], ase = sqrt(alpha.total$Q/n.obs),
##                 alpha.total[10])
##             colnames(alpha.total) <- c("raw_alpha", "std.alpha",
##                 "G6(smc)", "average_r", "S/N", "ase", "median_r")
##         }
##         rownames(alpha.total) <- ""
##         stats <- data.frame(r = item.r, r.cor = item.rc, r.drop = r.drop)
##     }
##     rownames(stats) <- colnames(x)
##     if (n.iter > 1) {
##         if (!raw) {
##             message("bootstrapped confidence intervals require raw data")
##             boot <- NULL
##             boot.ci <- NULL
##         }
##         else {
##             boot <- vector("list", n.iter)
##             boot <- mclapply(1:n.iter, function(XX) {
##                 xi <- x[sample.int(nsub, replace = TRUE), ]
##                 C <- cov(xi, use = "pairwise")
##                 if (!is.null(keys)) {
##                   key.d <- diag(keys)
##                   xi <- key.d %*% C %*% key.d
##                 }
##                 R <- cov2cor(C)
##                 alpha.1(C, R)
##             })
##             boot <- matrix(unlist(boot), ncol = 10, byrow = TRUE)
##             colnames(boot) <- c("raw_alpha", "std.alpha", "G6(smc)",
##                 "average_r", "s/n", "ase", "Unidim", "Goodfit",
```

```
##                  "var.r", "median.r")
##              boot.ci <- quantile(boot[, 1], c(0.025, 0.5, 0.975))
##          }
##      }
##      else {
##          boot = NULL
##          boot.ci <- NULL
##      }
##      names(Unidim) <- "Unidim"
##      names(Fit.off) <- "Fit.off"
##      result <- list(total = alpha.total, alpha.drop = by.item,
##          item.stats = stats, response.freq = response.freq, keys = keys,
##          scores = total, nvar = nvar, boot.ci = boot.ci, boot = boot,
##          Unidim = Unidim, var.r = var.r, Fit = Fit.off, call = cl,
##          title = title)
##      class(result) <- c("psych", "alpha")
##      return(result)
## }
## <bytecode: 0x7fad83e020c0>
## <environment: namespace:psych>
```

Missing data

```
# Missing data is represented in R by NA
x <- c(1, 2, NA, 4)
y <- c("a", "b", NA, "c")
x
```

```
## [1]  1  2 NA  4
```

```
y
```

```
## [1] "a" "b" NA  "c"
```

```
# To see whether a value is missing
is.na(x)
```

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

```
# If you have missing data, some functions will return NA by default
# rather than returning a value
mean(x)
```

```
## [1] NA
```

```
sd(x)
```

```
## [1] NA
```

```
# Many functions have a na.rm argument
# rm stands for "remove"
mean(x, na.rm = TRUE)
```

```
## [1] 2.333333
```

```
sd(x, na.rm = TRUE)
```

```
## [1] 1.527525
```

```r
# or you remove the missing data
na.omit(x)
```

```
## [1] 1 2 4
## attr(,"na.action")
## [1] 3
## attr(,"class")
## [1] "omit"
```

```r
mean(na.omit(x))
```

```
## [1] 2.333333
```

```r
# na.omit also works on data frames performing listwise deletion
head(survey)
```

```
##      Sex Wr.Hnd NW.Hnd W.Hnd    Fold Pulse    Clap Exer Smoke Height
## 1 Female   18.5   18.0 Right  R on L    92    Left Some Never 173.00
## 2   Male   19.5   20.5  Left  R on L   104    Left None Regul 177.80
## 3   Male   18.0   13.3 Right  L on R    87 Neither None Occas     NA
## 4   Male   18.8   18.9 Right  R on L    NA Neither None Never 160.00
## 5   Male   20.0   20.0 Right Neither    35   Right Some Never 165.00
## 6 Female   18.0   17.7 Right  L on R    64   Right Some Never 172.72
##        M.I    Age
## 1   Metric 18.250
## 2 Imperial 17.583
## 3     <NA> 16.917
## 4   Metric 20.333
## 5   Metric 23.667
## 6 Imperial 21.000
```

```r
dim(survey)
```

```
## [1] 237  12
```

```r
cleaned_survey <- na.omit(survey)
dim(cleaned_survey)
```

```
## [1] 168  12
```

Getting summaries of data frames

```r
library(MASS) # user survey data from MASS package
data(survey) # load an internal dataset
mydata <- survey

# Variable Names
names(mydata)
```

```
##  [1] "Sex"    "Wr.Hnd" "NW.Hnd" "W.Hnd"  "Fold"   "Pulse"  "Clap"
##  [8] "Exer"   "Smoke"  "Height" "M.I"    "Age"
```

```r
# Show structure
str(mydata)
```

```
## 'data.frame':    237 obs. of  12 variables:
```

```
##  $ Sex   : Factor w/ 2 levels "Female","Male": 1 2 2 2 2 1 2 1 2 2 ...
##  $ Wr.Hnd: num  18.5 19.5 18 18.8 20 18 17.7 17 20 18.5 ...
##  $ NW.Hnd: num  18 20.5 13.3 18.9 20 17.7 17.7 17.3 19.5 18.5 ...
##  $ W.Hnd : Factor w/ 2 levels "Left","Right": 2 1 2 2 2 2 2 2 2 2 ...
##  $ Fold  : Factor w/ 3 levels "L on R","Neither",..: 3 3 1 3 2 1 1 3 3 3 ...
##  $ Pulse : int  92 104 87 NA 35 64 83 74 72 90 ...
##  $ Clap  : Factor w/ 3 levels "Left","Neither",..: 1 1 2 2 3 3 3 3 3 3 ...
##  $ Exer  : Factor w/ 3 levels "Freq","None",..: 3 2 2 2 3 3 1 1 3 3 ...
##  $ Smoke : Factor w/ 4 levels "Heavy","Never",..: 2 4 3 2 2 2 2 2 2 2 ...
##  $ Height: num  173 178 NA 160 165 ...
##  $ M.I   : Factor w/ 2 levels "Imperial","Metric": 2 1 NA 2 2 1 1 2 2 2 ...
##  $ Age   : num  18.2 17.6 16.9 20.3 23.7 ...
```

```
# Useful summary of numeric and categorical variables
Hmisc::describe(mydata)
```

```
## mydata
##
##  12  Variables      237  Observations
## ---------------------------------------------------------------------------
## Sex
##        n  missing distinct
##      236        1        2
##
## Value      Female   Male
## Frequency     118    118
## Proportion    0.5    0.5
## ---------------------------------------------------------------------------
## Wr.Hnd
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      236        1       60    0.997    18.67     2.09    16.00    16.50
##      .25      .50      .75      .90      .95
##    17.50    18.50    19.80    21.15    22.05
##
## lowest : 13.0 14.0 15.0 15.4 15.5, highest: 22.5 22.8 23.0 23.1 23.2
## ---------------------------------------------------------------------------
## NW.Hnd
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      236        1       68    0.998    18.58    2.184    15.50    16.30
##      .25      .50      .75      .90      .95
##    17.50    18.50    19.72    21.00    22.22
##
## lowest : 12.5 13.0 13.3 13.5 15.0, highest: 22.7 23.0 23.2 23.3 23.5
## ---------------------------------------------------------------------------
## W.Hnd
##        n  missing distinct
##      236        1        2
##
## Value       Left Right
## Frequency     18   218
## Proportion 0.076 0.924
## ---------------------------------------------------------------------------
## Fold
##        n  missing distinct
##      237        0        3
```

```
##
## Value       L on R Neither  R on L
## Frequency       99      18     120
## Proportion   0.418   0.076   0.506
## ---------------------------------------------------------------------------
## Pulse
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      192       45       43    0.997    74.15    13.07    59.55    60.00
##      .25      .50      .75      .90      .95
##    66.00    72.50    80.00    90.00    92.00
##
## lowest :  35  40  48  50  54, highest:  96  97  98 100 104
## ---------------------------------------------------------------------------
## Clap
##        n  missing distinct
##      236        1        3
##
## Value         Left Neither   Right
## Frequency       39      50     147
## Proportion   0.165   0.212   0.623
## ---------------------------------------------------------------------------
## Exer
##        n  missing distinct
##      237        0        3
##
## Value       Freq  None  Some
## Frequency    115    24    98
## Proportion 0.485 0.101 0.414
## ---------------------------------------------------------------------------
## Smoke
##        n  missing distinct
##      236        1        4
##
## Value      Heavy Never Occas Regul
## Frequency     11   189    19    17
## Proportion 0.047 0.801 0.081 0.072
## ---------------------------------------------------------------------------
## Height
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      209       28       67    0.999    172.4     11.2    157.0    160.0
##      .25      .50      .75      .90      .95
##    165.0    171.0    180.0    185.4    189.6
##
## lowest : 150.00 152.00 152.40 153.50 154.94, highest: 191.80 193.04 195.00 196.00 200.00
## ---------------------------------------------------------------------------
## M.I
##        n  missing distinct
##      209       28        2
##
## Value      Imperial   Metric
## Frequency        68      141
## Proportion    0.325    0.675
## ---------------------------------------------------------------------------
## Age
```

```
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      237        0       88    0.999    20.37    4.353    17.08    17.22
##      .25      .50      .75      .90      .95
##    17.67    18.58    20.17    23.58    30.68
##
## lowest : 16.750 16.917 17.000 17.083 17.167, highest: 41.583 43.833 44.250 70.417 73.000
## ---------------------------------------------------------------------------
```

```
# Common univariate statistics for numeric variables
psych::describe(mydata)
```

```
##         vars   n   mean    sd median trimmed   mad    min   max range  skew
## Sex*       1 236   1.50  0.50   1.50    1.50  0.74   1.00   2.0  1.00  0.00
## Wr.Hnd     2 236  18.67  1.88  18.50   18.61  1.48  13.00  23.2 10.20  0.18
## NW.Hnd     3 236  18.58  1.97  18.50   18.55  1.63  12.50  23.5 11.00  0.02
## W.Hnd*     4 236   1.92  0.27   2.00    2.00  0.00   1.00   2.0  1.00 -3.17
## Fold*      5 237   2.09  0.96   3.00    2.11  0.00   1.00   3.0  2.00 -0.18
## Pulse      6 192  74.15 11.69  72.50   74.02 11.12  35.00 104.0 69.00 -0.02
## Clap*      7 236   2.46  0.76   3.00    2.57  0.00   1.00   3.0  2.00 -0.98
## Exer*      8 237   1.93  0.95   2.00    1.91  1.48   1.00   3.0  2.00  0.14
## Smoke*     9 236   2.18  0.62   2.00    2.07  0.00   1.00   4.0  3.00  1.67
## Height    10 209 172.38  9.85 171.00  172.19 10.08 150.00 200.0 50.00  0.22
## M.I*      11 209   1.67  0.47   2.00    1.72  0.00   1.00   2.0  1.00 -0.74
## Age       12 237  20.37  6.47  18.58   18.99  1.61  16.75  73.0 56.25  5.16
##         kurtosis   se
## Sex*       -2.01 0.03
## Wr.Hnd      0.30 0.12
## NW.Hnd      0.44 0.13
## W.Hnd*      8.10 0.02
## Fold*      -1.89 0.06
## Pulse       0.33 0.84
## Clap*      -0.60 0.05
## Exer*      -1.88 0.06
## Smoke*      3.21 0.04
## Height     -0.44 0.68
## M.I*       -1.46 0.03
## Age        33.47 0.42
```

Summaries of numeric vectors (or data frame variables)

```
x <- c(1, 2, 3, 4,5)

# Total
sum(x) # sum of vector
```

```
## [1] 15
```

```
length(x) # length of vector (e.g., sample size)
```

```
## [1] 5
```

```
# Central tendency
mean(x) # mean of vector
```

```
## [1] 3
```

```r
median(x) # median of vector
```

```
## [1] 3
```

```r
# Spread
sd(x) # standard deviation
```

```
## [1] 1.581139
```

```r
var(x) # variance
```

```
## [1] 2.5
```

```r
range(x) # min and max of vector
```

```
## [1] 1 5
```

```r
min(x) # minimum of vector
```

```
## [1] 1
```

```r
max(x) # max of vector
```

```
## [1] 5
```

```r
# Other distributional features
psych::skew(x) # skewness
```

```
## [1] 0
```

```r
psych::kurtosi(x) # kurtosis
```

```
## [1] -1.912
```

```r
dat <- data.frame(x = c(1, 2, 3, 4, 5),
                  y = c(0, 0, 1, 1, 1))
dat
```

```
##   x y
## 1 1 0
## 2 2 0
## 3 3 1
## 4 4 1
## 5 5 1
```

```r
# Vector operations typically operate element wise
dat$z <-  dat$x + dat$y
dat
```

```
##   x y z
## 1 1 0 1
## 2 2 0 2
## 3 3 1 4
## 4 4 1 5
## 5 5 1 6
```

```r
dat$z <-  dat$x * dat$y
dat
```

```
##   x y z
## 1 1 0 0
```

```
## 2 2 0 0
## 3 3 1 3
## 4 4 1 4
## 5 5 1 5
```

```r
# A single value is recyled through the vector
dat$z <- dat$x + 10
dat
```

```
##   x y  z
## 1 1 0 11
## 2 2 0 12
## 3 3 1 13
## 4 4 1 14
## 5 5 1 15
```

Exercise 2 - Data.frames

```r
# For this exercise will use the GSS7402 dataset
library(AER)
help(package = AER)
data("GSS7402")
?GSS7402 # to learn about the dataset
# It might be easier to work with a shorter variable name
gss <- GSS7402

# 1. List the variable names in the gss dataset

# 2. Show the first few rows (hint: the head) of the dataset?

# 3. How many cases are there?

# 4. What is the mean, sd, and range age of the sample

# 5. Use the psych and Hmisc describe functions to describe the samples

# 6. Extract a data.frame with only people over the age of 80

# 7. Get the mean number of children ("kids") for participants
#    over the age of 80

# 8. Use the mean function to get the mean age at first birth.
#    Hint: there is missing data.
```

Answers Exercise 2 - Data.frames

```r
# For this exercise will use the GSS7402 dataset
library(AER)
help(package = AER)
data("GSS7402")
?GSS7402 # to learn about the dataset
```

```r
# It might be easier to work with a shorter variable name
gss <- GSS7402

# 1. List the variable names in the gss dataset
names(gss)
```

```
##  [1] "kids"           "age"            "education"      "year"
##  [5] "siblings"       "agefirstbirth"  "ethnicity"      "city16"
##  [9] "lowincome16"    "immigrant"
```

```r
# 2. Show the first few rows (hint: the head) of the dataset?
head(gss)
```

```
##   kids age education year siblings agefirstbirth ethnicity city16
## 1    0  25        14 2002        1            NA      cauc     no
## 2    1  30        13 2002        4            19      cauc    yes
## 3    1  55         2 2002        1            27      cauc     no
## 4    2  57        16 2002        1            22      cauc     no
## 5    2  71        12 2002        6            29      cauc    yes
## 6    0  19        13 2002        1            NA     other    yes
##   lowincome16 immigrant
## 1          no        no
## 2          no        no
## 3          no       yes
## 4          no        no
## 5          no        no
## 6          no        no
```

```r
# 3. How many cases are there?
nrow(gss)
```

```
## [1] 9120
```

```r
# 4. What is the mean, sd, and range age of the sample
mean(gss$age)
```

```
## [1] 46.08202
```

```r
sd(gss$age)
```

```
## [1] 17.92389
```

```r
range(gss$age)
```

```
## [1] 18 89
```

```r
# 5. Use the psych and Hmisc describe functions to describe the samples
psych::describe(gss)
```

```
##                vars    n    mean    sd median trimmed   mad  min  max range
## kids              1 9120    2.08  1.81      2    1.86  1.48    0    8     8
## age               2 9120   46.08 17.92     43   44.94 19.27   18   89    71
## education         3 9120   12.64  2.96     12   12.70  2.97    0   20    20
## year              4 9120 1990.29  9.10   1994 1990.79 11.86 1974 2002    28
## siblings          5 9120    4.05  3.25      3    3.60  2.97    0   35    35
## agefirstbirth     6 3312   22.63  4.86     22   22.18  4.45    9   42    33
## ethnicity*        7 9120    1.80  0.40      2    1.88  0.00    1    2     1
## city16*           8 9120    1.42  0.49      1    1.41  0.00    1    2     1
```

```
## lowincome16*     9 9120    1.21  0.41      1    1.14  0.00    1    2     1
## immigrant*      10 9120    1.11  0.31      1    1.01  0.00    1    2     1
##                skew kurtosis   se
## kids           1.00     1.03 0.02
## age            0.48    -0.78 0.19
## education     -0.26     1.03 0.03
## year          -0.36    -1.16 0.10
## siblings       1.67     4.78 0.03
## agefirstbirth  0.87     0.59 0.08
## ethnicity*    -1.53     0.35 0.00
## city16*        0.30    -1.91 0.01
## lowincome16*   1.41    -0.02 0.00
## immigrant*     2.50     4.26 0.00
```

```
Hmisc::describe(gss)
```

```
## gss
##
##  10  Variables      9120  Observations
## ---------------------------------------------------------------------------
## kids
##        n  missing distinct     Info     Mean      Gmd
##     9120        0        9    0.961    2.076    1.941
##
## Value          0     1     2     3     4     5     6     7     8
## Frequency   2127  1544  2338  1474   790   376   208   100   163
## Proportion 0.233 0.169 0.256 0.162 0.087 0.041 0.023 0.011 0.018
## ---------------------------------------------------------------------------
## age
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     9120        0       72        1    46.08    20.38       22       25
##      .25      .50      .75      .90      .95
##       31       43       59       73       79
##
## lowest : 18 19 20 21 22, highest: 85 86 87 88 89
## ---------------------------------------------------------------------------
## education
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     9120        0       21    0.957    12.64    3.178        8        9
##      .25      .50      .75      .90      .95
##       12       12       14       16       18
##
## lowest :  0  1  2  3  4, highest: 16 17 18 19 20
## ---------------------------------------------------------------------------
## year
##        n  missing distinct     Info     Mean      Gmd
##     9120        0        8    0.979     1990     10.3
##
## Value       1974  1978  1982  1986  1990  1994  1998  2002
## Frequency    785   877  1064   842   767  1688  1580  1517
## Proportion 0.086 0.096 0.117 0.092 0.084 0.185 0.173 0.166
## ---------------------------------------------------------------------------
## siblings
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     9120        0       27    0.984    4.051    3.359        1        1
```

```
##      .25      .50      .75      .90      .95
##        2        3        6        8       10
##
## lowest :  0  1  2  3  4, highest: 22 23 25 27 35
## --------------------------------------------------------------------------
## agefirstbirth
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     3312     5808       33    0.995    22.63    5.345       16       17
##      .25      .50      .75      .90      .95
##       19       22       25       30       32
##
## lowest :  9 11 12 13 14, highest: 38 39 40 41 42
## --------------------------------------------------------------------------
## ethnicity
##        n  missing distinct
##     9120        0        2
##
## Value      other  cauc
## Frequency   1785  7335
## Proportion 0.196 0.804
## --------------------------------------------------------------------------
## city16
##        n  missing distinct
##     9120        0        2
##
## Value         no   yes
## Frequency   5246  3874
## Proportion 0.575 0.425
## --------------------------------------------------------------------------
## lowincome16
##        n  missing distinct
##     9120        0        2
##
## Value         no   yes
## Frequency   7182  1938
## Proportion 0.788 0.212
## --------------------------------------------------------------------------
## immigrant
##        n  missing distinct
##     9120        0        2
##
## Value         no   yes
## Frequency   8122   998
## Proportion 0.891 0.109
## --------------------------------------------------------------------------
```

```r
# 6. Extract a data.frame with only people over the age of 80
gss_over80 <- gss[ gss$age > 80, ]


# 7. Get the mean number of children ("kids") for participants
#    over the age of 80
mean(gss[ gss$age > 80, "kids"])
```

```
## [1] 2.394737
```

```r
# 8. Use the mean function to get the mean age at first birth.
#    Hint: there is missing data.
mean(gss$agefirstbirth) # doesn't work because there is missing data
```

```
## [1] NA
```

```r
mean(gss$agefirstbirth, na.rm = TRUE) # doesn't work because there is missing data
```

```
## [1] 22.63074
```

String functions

```r
paste("hello", "how", "are", "You") # defaults to space separator
```

```
## [1] "hello how are You"
```

```r
paste0("hello", "how", "are", "You") # no separator
```

```
## [1] "hellohowareYou"
```

```r
paste("apple", "banana", "carrot", "date", sep =", ") # specify arbitrary separator
```

```
## [1] "apple, banana, carrot, date"
```

```r
paste0("v", 1:10) # paticularly useful with vectors
```

```
##  [1] "v1"  "v2"  "v3"  "v4"  "v5"  "v6"  "v7"  "v8"  "v9"  "v10"
```

```r
# Extract substring
substr("abcdefghijklmnop", 4, 6)
```

```
## [1] "def"
```

```r
# Change case
toupper("abcd") # make upper case
```

```
## [1] "ABCD"
```

```r
tolower("ABCD") # make lower case
```

```
## [1] "abcd"
```

```r
mystring <- c("apple", "banana", "carrot", "date", "egg", "fig")
# Identify which strings match a pattern
grep("a", mystring) # index of objects with "a"
```

```
## [1] 1 2 3 4
```

```r
grep("a", mystring, value = TRUE) # value of objects with "a"
```

```
## [1] "apple"  "banana" "carrot" "date"
```

```r
# get count of number of characters
nchar(mystring)
```

```
## [1] 5 6 6 4 3 3
```

```r
data.frame(mystring, nchar(mystring))
```

```
##   mystring nchar.mystring.
```

```
## 1    apple                5
## 2   banana                6
## 3   carrot                6
## 4     date                4
## 5      egg                3
## 6      fig                3
```

```
# Substitute a mystringreplacement text that matches a pattern
questions <- c("How are you?", "What is going on?")
gsub(" ", "_", questions) # replace space with underscore
```

```
## [1] "How_are_you?"      "What_is_going_on?"
```

```
# R  string manipulation tools are very powerful
# For more information see
?grep
?"regular expression"


# see also Hadley Wickham's package for string manipulation
# It attempts to introduce greater consistency in notation.
# install.packages("stringr")
library(stringr)
```

```
## Warning: package 'stringr' was built under R version 3.5.2
```

```
help(package = "stringr")
# all functions begin with str_
str_length(mystring) # see nchar
```

```
## [1] 5 6 6 4 3 3
```

```
str_sub(mystring, start = 1, end = 3)
```

```
## [1] "app" "ban" "car" "dat" "egg" "fig"
```

```
# writing output to the console
cat("Hello World!")
```

```
## Hello World!
```

```
# Tab is \t and new line is \n
cat("Hello\t World\nSome more text")
```

```
## Hello     World
## Some more text
```

Importing data

```
# A simple option is to export data from your external data
# in csv format and then import the data using csv
# csv
medals <- read.csv("data/practice/medals.csv")
head(medals)
```

```
##    Year     City     Sport     Discipline NOC          Event Event.gender
## 1 1924 Chamonix    Skating Figure skating AUT      individual            M
```

```
## 2 1924 Chamonix    Skating Figure skating AUT      individual             W
## 3 1924 Chamonix    Skating Figure skating AUT           pairs             X
## 4 1924 Chamonix  Bobsleigh      Bobsleigh BEL        four-man             M
## 5 1924 Chamonix Ice Hockey     Ice Hockey CAN      ice hockey             M
## 6 1924 Chamonix   Biathlon       Biathlon FIN military patrol             M
##    Medal
## 1 Silver
## 2   Gold
## 3   Gold
## 4 Bronze
## 5   Gold
## 6 Silver
```

```r
tail(medals)
```

```
##      Year  City  Sport Discipline NOC          Event Event.gender  Medal
## 2306 2006 Turin Skiing  Snowboard USA       Half-pipe            M   Gold
## 2307 2006 Turin Skiing  Snowboard USA       Half-pipe            M Silver
## 2308 2006 Turin Skiing  Snowboard USA       Half-pipe            W   Gold
## 2309 2006 Turin Skiing  Snowboard USA       Half-pipe            W Silver
## 2310 2006 Turin Skiing  Snowboard USA Snowboard Cross            M   Gold
## 2311 2006 Turin Skiing  Snowboard USA Snowboard Cross            W Silver
```

```r
dim(medals)
```

```
## [1] 2311    8
```

```r
# Other delimited formats
medals <- read.table("data/practice/medals.tsv", sep ="\t")

# Read Excel
library(readxl)
```

```
## Warning: package 'readxl' was built under R version 3.5.2
```

```r
# note that readxl returns a tibble,
# which is almost but not quite the same as a data.frame
medals <- data.frame(readxl::read_excel("data/practice/medals.xls"))
head(medals)
```

```
##   Year     City      Sport     Discipline NOC           Event Event.gender
## 1 1924 Chamonix    Skating Figure skating AUT      individual             M
## 2 1924 Chamonix    Skating Figure skating AUT      individual             W
## 3 1924 Chamonix    Skating Figure skating AUT           pairs             X
## 4 1924 Chamonix  Bobsleigh      Bobsleigh BEL        four-man             M
## 5 1924 Chamonix Ice Hockey     Ice Hockey CAN      ice hockey             M
## 6 1924 Chamonix   Biathlon       Biathlon FIN military patrol             M
##    Medal
## 1 Silver
## 2   Gold
## 3   Gold
## 4 Bronze
## 5   Gold
## 6 Silver
```

```r
# SPSS
library(foreign)
cas <- foreign::read.spss("data/practice/cas.sav", to.data.frame = TRUE)
```

```r
attr(cas, "variable.labels")
```

```
##                                                 district
##                                          "District code"
##                                                   school
##                                            "School name"
##                                                   county
##                                                 "County"
##                                                   grades
##                                 "grade span of district"
##                                                 students
##                                       "Total enrollment"
##                                                 teachers
##                                     "Number of teachers"
##                                                 calworks
## "Percent qualifying for CalWorks (income assistance)"
##                                                    lunch
##          "Percent qualifying for reduced-price lunch"
##                                                 computer
##                                    "Number of computers"
##                                              expenditure
##                                "Expenditure per student"
##                                                   income
##              "District average income (in USD 1,000)"
##                                                  english
##                             "Percent of English learners"
##                                                     read
##                                  "Average reading score"
##                                                     math
##                                     "Average math score"
```

```r
# tip: You may need to think about value labels in your SPSS file
# Specifically, if you have numeric variables that have variable labels, you may
# want to remove the value labels in SPSS or

# import stata, sas
?read.dta
?read.sas
```

```
## No documentation for 'read.sas' in specified packages and libraries:
## you could try '??read.sas'
```

```r
# Use ProjectTemplate to auto-import (see discussion later)
```

Exporting data

```r
mydata <- data.frame(a = c(1,2,3), b = c("a", "b", "c"))

# Interal R format
# Good option if you need to re-open data in R
save(mydata, file="output/mydata.rdata")
#  load("output/mydata.rdata")
```

```r
# csv
# Good option if you need to get data into other software
# This should open in almost all other software (e.g. Excel, SPSS, etc.)
write.csv(mydata, file = "output/mydata.csv")
write.csv(mydata, file = "output/mydata-2.csv", row.names = FALSE) # exclude row.names

# If you need more flexibility in terms of delimiters, etc.
write.table(mydata, file = "output/mydata.tsv", sep = "\t") # e.g., tab delimiter

# write excel
library(openxlsx)
```

```
## Warning: package 'openxlsx' was built under R version 3.5.2
```

```r
openxlsx::write.xlsx(mydata, file = "output/mydata.xlsx")
```

```
## Note: zip::zip() is deprecated, please use zip::zipr() instead
```

```r
# Exporting to other formats
# There are a range of options for exporting to other formats
# Functionality is often spread around
# Given that the csv option is usually sufficient
library(foreign)
?foreign::write.foreign  # options for exporting to SAS, SPSS, and Stata directly
```

Exercise 3

```r
# 1. Open medals.csv in the data/practice/ directory
#    and assign to variable medals

# 2. Check that the file imported correctled
#    (a) look at the first few rows,
#    (b) look at the last few rows,
#    (b) check the structure (i.e., str),
#    (c) Use the Hmisc describe function to check basic properties

# 3. Create a new variable in medals that indicates
#    whether the medals was Gold (TRUE) or Silver/Bronze (FALSE)
#    and call it isgold

# 4. Calculate the total number of gold medals

# 5. Export the medals data.frame to the output folder
#    (a) as a csv file
#    (b) as a native rdata file

# 6. Remove the medals dataset from the workspace
#    and then load it again from the csv file.
#    Check that it imported correctly.
# Then remove medals and repeat for the rdata file
```

Answers for Exercise 3

```
# 1. Open medals.csv in the data/practice/ directory
#    and assign to variable medals
medals <- read.csv("data/practice/medals.csv")

# 2. Check that the file imported correctled
#    (a) look at the first few rows,
#    (b) look at the last few rows,
#    (b) check the structure (i.e., str),
#    (c) Use the Hmisc describe function to check basic properties
head(medals)
```

```
##   Year     City      Sport     Discipline NOC           Event Event.gender
## 1 1924 Chamonix    Skating Figure skating AUT      individual            M
## 2 1924 Chamonix    Skating Figure skating AUT      individual            W
## 3 1924 Chamonix    Skating Figure skating AUT           pairs            X
## 4 1924 Chamonix  Bobsleigh      Bobsleigh BEL        four-man            M
## 5 1924 Chamonix Ice Hockey     Ice Hockey CAN      ice hockey            M
## 6 1924 Chamonix   Biathlon       Biathlon FIN military patrol            M
##    Medal
## 1 Silver
## 2   Gold
## 3   Gold
## 4 Bronze
## 5   Gold
## 6 Silver
```

```
tail(medals)
```

```
##      Year  City  Sport Discipline NOC          Event Event.gender  Medal
## 2306 2006 Turin Skiing  Snowboard USA       Half-pipe            M   Gold
## 2307 2006 Turin Skiing  Snowboard USA       Half-pipe            M Silver
## 2308 2006 Turin Skiing  Snowboard USA       Half-pipe            W   Gold
## 2309 2006 Turin Skiing  Snowboard USA       Half-pipe            W Silver
## 2310 2006 Turin Skiing  Snowboard USA Snowboard Cross            M   Gold
## 2311 2006 Turin Skiing  Snowboard USA Snowboard Cross            W Silver
```

```
str(medals)
```

```
## 'data.frame':    2311 obs. of  8 variables:
##  $ Year        : int  1924 1924 1924 1924 1924 1924 1924 1924 1924 1924 ...
##  $ City        : chr  "Chamonix" "Chamonix" "Chamonix" "Chamonix" ...
##  $ Sport       : chr  "Skating" "Skating" "Skating" "Bobsleigh" ...
##  $ Discipline  : chr  "Figure skating" "Figure skating" "Figure skating" "Bobsleigh" ...
##  $ NOC         : chr  "AUT" "AUT" "AUT" "BEL" ...
##  $ Event       : chr  "individual" "individual" "pairs" "four-man" ...
##  $ Event.gender: chr  "M" "W" "X" "M" ...
##  $ Medal       : chr  "Silver" "Gold" "Gold" "Bronze" ...
```

```
Hmisc::describe(medals)
```

```
## medals
##
##  8  Variables      2311  Observations
## ---------------------------------------------------------------------------
```

```
## Year
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     2311        0       20    0.995     1980    24.32     1932     1948
##      .25      .50      .75      .90      .95
##     1968     1988     1998     2006     2006
##
## Value       1924  1928  1932  1936  1948  1952  1956  1960  1964  1968
## Frequency     49    41    42    51    68    67    72    81   103   106
## Proportion 0.021 0.018 0.018 0.022 0.029 0.029 0.031 0.035 0.045 0.046
##
## Value       1972  1976  1980  1984  1988  1992  1994  1998  2002  2006
## Frequency    105   111   115   117   138   171   183   205   234   252
## Proportion 0.045 0.048 0.050 0.051 0.060 0.074 0.079 0.089 0.101 0.109
## ---------------------------------------------------------------------------
## City
##        n  missing distinct
##     2311        0       17
##
## Albertville (171, 0.074), Calgary (138, 0.060), Chamonix (49, 0.021),
## Cortina d'Ampezzo (72, 0.031), Garmisch-Partenkirchen (51, 0.022),
## Grenoble (106, 0.046), Innsbruck (214, 0.093), Lake Placid (157, 0.068),
## Lillehammer (183, 0.079), Nagano (205, 0.089), Oslo (67, 0.029), Salt Lake
## City (234, 0.101), Sapporo (105, 0.045), Sarajevo (117, 0.051), Squaw
## Valley (81, 0.035), St. Moritz (109, 0.047), Turin (252, 0.109)
## ---------------------------------------------------------------------------
## Sport
##        n  missing distinct
##     2311        0        7
##
## Value        Biathlon  Bobsleigh    Curling Ice Hockey       Luge
## Frequency         162        133         21         69        108
## Proportion      0.070      0.058      0.009      0.030      0.047
##
## Value         Skating     Skiing
## Frequency         758       1060
## Proportion      0.328      0.459
## ---------------------------------------------------------------------------
## Discipline
##        n  missing distinct
##     2311        0       15
##
## Alpine Skiing (367, 0.159), Biathlon (162, 0.070), Bobsleigh (115, 0.050),
## Cross Country S (399, 0.173), Curling (21, 0.009), Figure skating (207,
## 0.090), Freestyle Ski. (54, 0.023), Ice Hockey (69, 0.030), Luge (108,
## 0.047), Nordic Combined (84, 0.036), Short Track S. (96, 0.042), Skeleton
## (18, 0.008), Ski Jumping (114, 0.049), Snowboard (42, 0.018), Speed
## skating (455, 0.197)
## ---------------------------------------------------------------------------
## NOC
##        n  missing distinct
##     2311        0       45
##
## lowest : AUS AUT BEL BLR BUL, highest: UKR URS USA UZB YUG
## ---------------------------------------------------------------------------
```

```
## Event
##        n  missing distinct
##     2311        0       67
##
## lowest : 10000m            1000m             10km              10km pursuit      12,5km mass start
## highest: super-G           Team              Team pursuit      Team sprint       two-man
## ---------------------------------------------------------------------------
## Event.gender
##        n  missing distinct
##     2311        0        3
##
## Value          M     W     X
## Frequency   1386   802   123
## Proportion 0.600 0.347 0.053
## ---------------------------------------------------------------------------
## Medal
##        n  missing distinct
##     2311        0        3
##
## Value      Bronze   Gold Silver
## Frequency     764    774    773
## Proportion  0.331  0.335  0.334
## ---------------------------------------------------------------------------
```

```r
# 3. Create a new variable in medals that indicates
#    whether the medals was Gold (TRUE) or Silver/Bronze (FALSE)
#    and call it isgold
medals$isgold <- medals$Medal == "Gold"

# 4. Calculate the total number of gold medals
sum(medals$isgold)
```

```
## [1] 774
```

```r
# 5. Export the medals data.frame to the output folder
#    (a) as a csv file
#    (b) as a native rdata file
write.csv(medals, "output/medals.csv")
# or technically you may want to do
write.csv(medals, "output/medals.csv", row.names = FALSE)
save(medals, file = "output/medals.rdata")

# 6. Remove the medals dataset from the workspace
#    and then load it again from the csv file.
#    Check that it imported correctly.
# Then remove medals and repeat for the rdata file
rm(medals)
medals <- read.csv("output/medals.csv")
head(medals)
```

```
##   Year     City      Sport     Discipline NOC          Event Event.gender
## 1 1924 Chamonix    Skating Figure skating AUT      individual            M
## 2 1924 Chamonix    Skating Figure skating AUT      individual            W
## 3 1924 Chamonix    Skating Figure skating AUT           pairs            X
## 4 1924 Chamonix  Bobsleigh      Bobsleigh BEL       four-man            M
## 5 1924 Chamonix Ice Hockey     Ice Hockey CAN      ice hockey            M
```

```
## 6 1924 Chamonix   Biathlon       Biathlon FIN military patrol            M
##    Medal isgold
## 1 Silver  FALSE
## 2   Gold   TRUE
## 3   Gold   TRUE
## 4 Bronze  FALSE
## 5   Gold   TRUE
## 6 Silver  FALSE
```

```r
rm(medals)
load("output/medals.rdata")
head(medals)
```

```
##   Year     City      Sport     Discipline NOC          Event Event.gender
## 1 1924 Chamonix    Skating Figure skating AUT     individual            M
## 2 1924 Chamonix    Skating Figure skating AUT     individual            W
## 3 1924 Chamonix    Skating Figure skating AUT          pairs            X
## 4 1924 Chamonix  Bobsleigh      Bobsleigh BEL       four-man            M
## 5 1924 Chamonix Ice Hockey     Ice Hockey CAN     ice hockey            M
## 6 1924 Chamonix   Biathlon       Biathlon FIN military patrol            M
##    Medal isgold
## 1 Silver  FALSE
## 2   Gold   TRUE
## 3   Gold   TRUE
## 4 Bronze  FALSE
## 5   Gold   TRUE
## 6 Silver  FALSE
```

Random variables and distributions

```r
# In statistics, we often want to generate random data with certain properties
# or looking up features of statistical distributions.
# See the following help for list of common distributions is base R
?Distributions

# and see http://cran.r-project.org/web/views/Distributions.html for many more distributions

# Each distribution has four functions that differ in terms of the first letter
# For example, for the normal distribution, you have
dnorm(1) # Density of the value 1 of a standard normal distribution
```

```
## [1] 0.2419707
```

```r
pnorm(1) # Cumulative distribution function for value of 1 on standard normal distribution
```

```
## [1] 0.8413447
```

```r
qnorm(.975) # Inverse cumulative distribution function for value of .975
```

```
## [1] 1.959964
```

```r
rnorm(5) # Generate 5 random draws from normal distribution
```

```
## [1] -1.428656760  0.008139295 -0.203252620 -2.308346829 -1.063667774
```

```r
#
dunif(1) # Density of the value 1 of a uniform distribution (0, 1)
```

```
## [1] 1
```

```r
punif(.5) # Cumulative distribution function for value of 1 on uniform distribution
```

```
## [1] 0.5
```

```r
qunif(.975) # Inverse cumulative distribution function for value of .975
```

```
## [1] 0.975
```

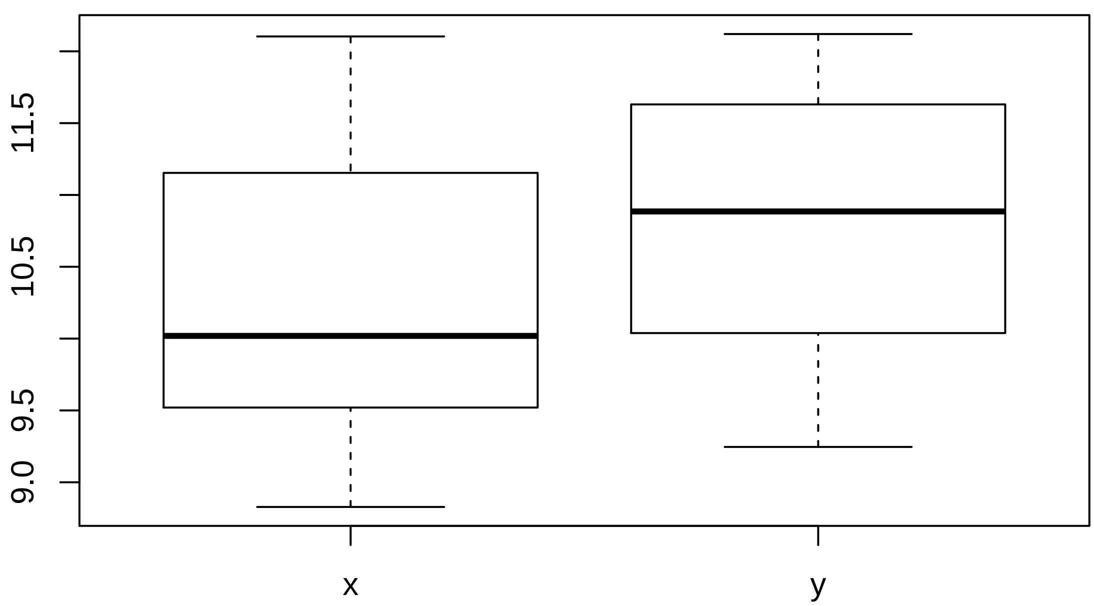
```r
runif(5) # Generate 5 random draws from uniform distribution
```

```
## [1] 0.7419174 0.0461430 0.7963427 0.3642507 0.2666787
```

```r
# Distributions have parameters that can be specified
x <- rnorm(10, mean = 10, sd = 1) # draw 10 from mean of 10
y <- rnorm(10, mean = 11, sd = 1) # draw 10 from mean of 11
dat <- data.frame(x=x, y=y)
dat
```

```
##            x         y
## 1  10.336496 11.020999
## 2   9.200158  9.497520
## 3   9.530175 11.630426
## 4   8.828091 11.267821
## 5  12.103554 12.120114
## 6   9.701318 10.749551
## 7   9.520142  9.245939
## 8  10.590699 10.725930
## 9  11.153680 11.937980
## 10 11.751954 10.038433
```

```r
boxplot(dat)
```



Functions

```r
# You can write functions and these are generally the same as
# the functions you use in R

# For example, I could create a function that printed some text
print_some_text <- function(x = "Hello World") {
    print(x)
}

# If I run the above command, I can then use it
print_some_text() # using the default argument
```

```
## [1] "Hello World"
```

```r
print_some_text("blah blah blah")  # or to print some other text
```

```
## [1] "blah blah blah"
```

```r
# Anatomy of a function
# Functions have a name
# They take one or more arguments
# Arguments may have default values


# Let's take a more interesting example: Power analysis
# The following data simulates data for two groups and
# examines whether there is a significant difference at .05
# It repeats the process 1000 times and calculates the
# proportion of times it is statistically significant
# (i.e., simluation estimate of the statistical power)

significant <- NULL
for (i in 1:1000) {
    x <- rnorm(30, mean = 0.0, sd = 1)
    y <- rnorm(30, mean = 0.3, sd = 1)
    fit <- t.test(x, y)
    fit
    significant[i] <- (fit$p.value < .05)
}
statistical_power <- mean(significant)
statistical_power
```

```
## [1] 0.203
```

```r
# we could convert this to a function
power_group_dif1 <- function() {
    significant <- NULL
    for (i in 1:1000) {
        x <- rnorm(30, mean = 0.0, sd = 1)
        y <- rnorm(30, mean = 0.3, sd = 1)
        fit <- t.test(x, y)
        fit
        significant[i] <- (fit$p.value < .05)
    }
```

```r
    statistical_power <- mean(significant)
    statistical_power
}

power_group_dif1()
```

```
## [1] 0.186
```

```r
# but the beauty of function is that they can make things general
# Let's make the mean of group 2 an argument that can be specified
power_group_dif2 <- function(mean2 = 0.3) {
    significant <- NULL
    for (i in 1:1000) {
        x <- rnorm(30, mean = 0.0, sd = 1)
        y <- rnorm(30, mean = mean2, sd = 1)
        fit <- t.test(x, y)
        fit
        significant[i] <- (fit$p.value < .05)
    }
    statistical_power <- mean(significant)
    statistical_power
}

# now we can specify different values
power_group_dif2(0)
```

```
## [1] 0.052
```

```r
power_group_dif2(.3)
```

```
## [1] 0.203
```

```r
power_group_dif2(.5)
```

```
## [1] 0.475
```

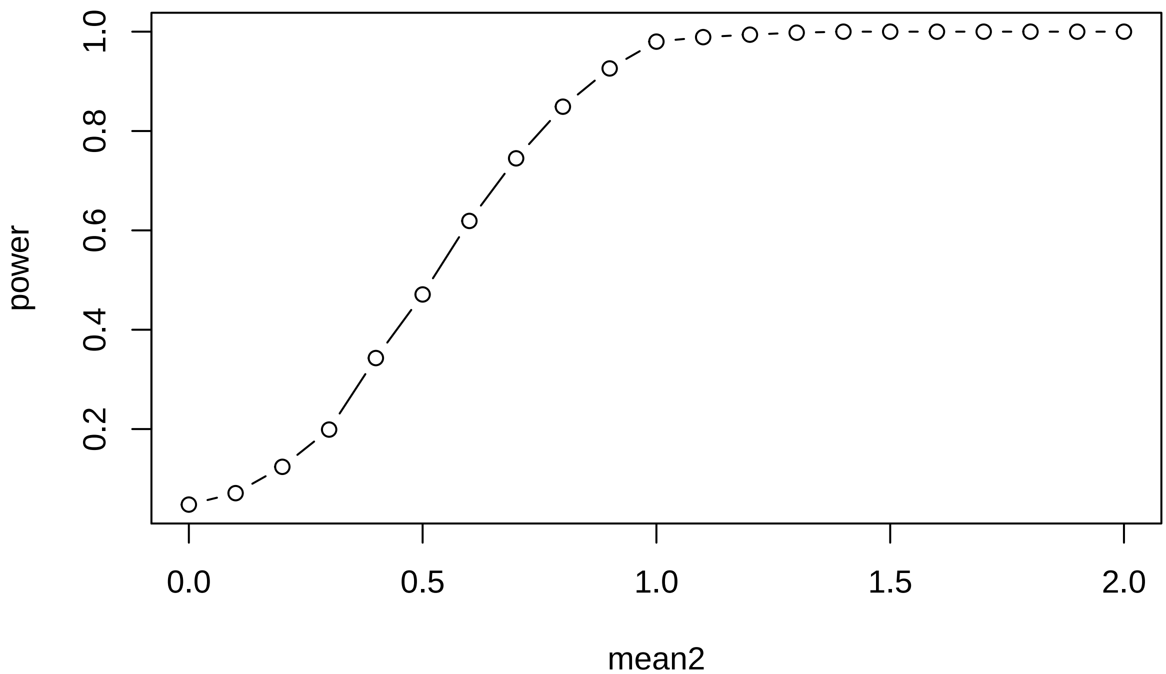
```r
power_group_dif2(.8)
```

```
## [1] 0.846
```

```r
power_group_dif2(1)
```

```
## [1] 0.97
```

```r
settings <- seq(from = 0, to = 2, by = .1)
results <- data.frame(mean2= settings)
results$power <- sapply(results$mean2, function(X) power_group_dif2(X))
plot(results, type = "b")
```




```r
# obviously it could be made a whole lot more general
power_group_dif3 <- function(mean1 = 0, mean2 = 0.3, sd1 = 1, sd2 = 1,
                             n1 = 30 , n2 = 30, ksimulations = 1000,
                             alpha_criterion = .05) {
    significant <- NULL
    for (i in 1:ksimulations) {
        x <- rnorm(30, mean = mean1, sd = sd1)
        y <- rnorm(30, mean = mean2, sd = sd2)
        fit <- t.test(x, y)
        fit
        significant[i] <- (fit$p.value < alpha_criterion)
    }
    statistical_power <- mean(significant)
    statistical_power
}

power_group_dif3(mean1 = 10, mean2 = 11, sd1 = 1, sd2 = 1,
                             n1 = 100 , n2 = 100, ksimulations = 10000,
                             alpha_criterion = .01)
```

```
## [1] 0.8867
```

Debugging functions

```r
# debugging functions
print_some_text <- function(x = "Hello World") {
    print(x)
}
debugonce(print_some_text) # activates debugging on the function
print_some_text()
```

```
## debugging in: print_some_text()
## debug at <text>#2: {
##     print(x)
## }
## debug at <text>#3: print(x)
## [1] "Hello World"
## exiting from: print_some_text()
```

```
# many other useful functions
?traceback # provide further information when an error occurs
?browser # place in function
```

Viewing source code for internal functions

```
# Option 1: type function name
t.test
cor
power.t.test


# Option 2:
# S3 Methods
# Some functions are generic and operate differently depending
# on the class of the first argument
# mean
# print
# summary

#  Methods will list the actual function names called
methods(mean)
methods(print)
methods(summary)

mean.default
summary.table


# Option 3:
# Some functions are part of packages but are not exported
# I.e., they are intended for internal use, but
# they are often quite useful
library(ProjectTemplate)
```

```
## Warning: package 'ProjectTemplate' was built under R version 3.5.2
```

```
# Double colon shows the functions exported from a package
# i.e., packagename::function
ProjectTemplate::run.project

# Triple colon shows internal functions
# i.e., packagename:::function
ProjectTemplate:::xls.reader
```

```r
# Also, see the getAnywhere function
xls.reader # this doesn't work
```

```
## Error in eval(expr, envir, enclos): object 'xls.reader' not found
```

```r
getAnywhere(xls.reader) # this does work
```

Exercise 4

```r
library(MASS)
data(mammals)
?mammals
head(mammals)
```

```
##                     body brain
## Arctic fox         3.385  44.5
## Owl monkey         0.480  15.5
## Mountain beaver    1.350   8.1
## Cow              465.000 423.0
## Grey wolf         36.330 119.5
## Goat              27.660 115.0
```

```r
# 1. Create a function that takes a single argument x
#    and prints that value twice.
#    use the function to print "hello world" twice


# 2. Divide mammall brain weight (g) by body weight (kg) and
#    get the mean of this value


# 3. Write a function that takes arguments x and y
#    and returns the mean of x divided by y


# 4. Apply the function to get the mean ratio of brain to body size


# 5. Modify the ratio function to return a list with
#    (a) the mean of x divided by y, and
#    (b) the sd of x divided by y.
#    Then apply to mammals data as above.


# 6. Step through the code for the correlation function


# 7. Show the source code for
#    (a) the t.test function,
#    (b) the summary method for lm objects
#    (c) the alpha function in the psych package
```

Answers 4

```r
library(MASS)
data(mammals)
?mammals
head(mammals)
```

```
##                    body brain
## Arctic fox        3.385  44.5
## Owl monkey        0.480  15.5
## Mountain beaver   1.350   8.1
## Cow             465.000 423.0
## Grey wolf        36.330 119.5
## Goat             27.660 115.0
```

```r
# 1. Create a function that takes a single argument x
#    and prints that value twice.
#    use the function to print "hello world" twice
print_twice <- function(x) {
    print(x)
    print(x)
}
print_twice("hello world")
```

```
## [1] "hello world"
## [1] "hello world"
```

```r
# 2. Divide mammall brain weight (g) by body weight (kg) and
#    get the mean of this value
mean(mammals$brain / mammals$body )
```

```
## [1] 9.624214
```

```r
# 3. Write a function that takes arguments x and y
#    and returns the mean of x divided by y
mean_ratio <- function(x, y) {
    mean(x / y)
}

# 4. Apply the function to get the mean ratio of brain to body size
mean_ratio(mammals$brain, mammals$body)
```

```
## [1] 9.624214
```

```r
# 5. Modify the ratio function to return a list with
#    (a) the mean of x divided by y, and
#    (b) the sd of x divided by y.
#    Then apply to mammals data as above.
mean_ratio <- function(x, y) {
    ratioxy <- x / y
    list(mean_ratio = mean(ratioxy),
         sd_ratio = sd(ratioxy))
}

# 6. Step through the code for the correlation function
# debugonce(cor)
```

```r
cor(mammals$brain, mammals$body, method = "spearman")
```

```
## [1] 0.9534986
```

```r
# 7. Show the source code for
#    (a) the t.test function,
#    (b) the summary method for lm objects
#    (c) the alpha function in the psych package
# t.test
# summary.lm
# psych::alpha
```