Introduction to R

2022-11-03

```
head(mtcars)
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
# Calculate 3 + 4
3 + 4
```

```
## [1] 7
```

```
# Calculate 6 + 12
6 + 12
```

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

```
# An addition
5 + 5
```

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

```
# A subtraction
5 - 5
```

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

```
# A multiplication
3 * 5
```

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

```
 # A division
(5 + 5) / 2
```

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

```r
# Exponentiation
2 ^ 5
```

```
## [1] 32
```

```r
# Modulo
28 %% 6
```

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

```r
# Assign the value 42 to x
x <- 42

# Print out the value of the variable x
print(x)
```

```
## [1] 42
```

```r
# Assign the value 5 to the variable my_apples
my_apples <- 5

# Print out the value of the variable my_apples
print(my_apples)
```

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

```r
# Assign a value to the variables my_apples and my_oranges
my_apples <- 5


# Add these two variables together
my_oranges <- 6

# Create the variable my_fruit
my_fruit <- (my_apples + my_oranges)
print(my_fruit)
```

```
## [1] 11
```

```r
# Declare variables of different types:
my_numeric <- 42
my_character <- "universe"
my_logical <- FALSE

# Check class of my_numeric
class(my_numeric)
```

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

```
# Check class of my_character
class(my_character)
```

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

```
# Check class of my_logical
class(my_logical)
```

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

```
# Define the variable vegas
vegas <- "Go!"
numeric_vector <- c(1, 10, 49)
character_vector <- c("a", "b", "c")

# Complete the code for boolean_vector
boolean_vector <- c(TRUE,FALSE,TRUE)
```

```
# Poker winnings from Monday to Friday
poker_vector <- c(140, -50, 20, -120, 240)

# Roulette winnings from Monday to Friday
roulette_vector <-  c(-24, -50, 100, -350, 10)

# The variable days_vector
days_vector <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday")

# Assign days as names of poker_vector
names(poker_vector) <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday")

# Assign days as names of roulette_vector
names(roulette_vector) <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday")
```

```
A_vector <- c(1, 2, 3)
B_vector <- c(4, 5, 6)

# Take the sum of A_vector and B_vector
total_vector <- A_vector + B_vector

# Print out total_vector
print(total_vector)
```

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

```
# Poker and roulette winnings from Monday to Friday:
poker_vector <- c(140, -50, 20, -120, 240)
roulette_vector <- c(-24, -50, 100, -350, 10)
days_vector <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday")
names(poker_vector) <- days_vector
names(roulette_vector) <- days_vector
```

```
# Assign to total_daily how much you won/lost on each day
total_daily <- poker_vector + roulette_vector
# Total winnings with poker
total_poker <- sum(poker_vector)

# Total winnings with roulette
total_roulette <-  sum(roulette_vector)

# Total winnings overall
total_week <- total_roulette + total_poker
# Check if you realized higher total gains in poker than in roulette
total_poker > total_roulette
```

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

```
# Print out total_week
print(total_week)
```

```
## [1] -84
```

```
# Define a new variable based on a selection
poker_wednesday <- 20

# Define a new variable based on a selection
poker_midweek <- poker_vector[c(2, 3, 4)]

# Define a new variable based on a selection
roulette_selection_vector <- roulette_vector[2:5]

# Select poker results for Monday, Tuesday and Wednesday
poker_start <- poker_vector[c("Monday", "Tuesday", "Wednesday")]

# Calculate the average of the elements in poker_start
mean(poker_start)
```

```
## [1] 36.66667
```

```
# Which days did you make money on poker?
selection_vector <- poker_vector > 0

# Print out selection_vector
selection_vector
```

```
##    Monday   Tuesday Wednesday  Thursday    Friday
##      TRUE     FALSE      TRUE     FALSE      TRUE
```

```
# Select from poker_vector these days
poker_winning_days <- poker_vector[selection_vector]

# Which days did you make money on roulette?
```

```
selection_vector <- roulette_vector > 0

# Select from roulette_vector these days
roulette_winning_days <- roulette_vector[selection_vector]
```

```
# Construct a matrix with 3 rows that contain the numbers 1 up to 9
matrix(1:9, byrow = TRUE, nrow = 3 )
```

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

```
# Box office Star Wars (in millions!)
new_hope <- c(460.998, 314.4)
empire_strikes <- c(290.475, 247.900)
return_jedi <- c(309.306, 165.8)

# Create box_office
box_office <- c(new_hope, empire_strikes, return_jedi)

# Construct star_wars_matrix
star_wars_matrix <- matrix(box_office, nrow = 3, byrow = TRUE)
```

```
# Box office Star Wars (in millions!)
new_hope <- c(460.998, 314.4)
empire_strikes <- c(290.475, 247.900)
return_jedi <- c(309.306, 165.8)

# Construct matrix
star_wars_matrix <- matrix(c(new_hope, empire_strikes, return_jedi), nrow = 3, byrow = TRUE)

# Vectors region and titles, used for naming
region <- c("US", "non-US")
titles <- c("A New Hope", "The Empire Strikes Back", "Return of the Jedi")

# Name the columns with region
colnames(star_wars_matrix) <- region
# Name the rows with titles
rownames(star_wars_matrix) <- titles
# Print out star_wars_matrix
star_wars_matrix
```

```
##                              US non-US
## A New Hope              460.998  314.4
## The Empire Strikes Back 290.475  247.9
## Return of the Jedi      309.306  165.8
```

```
# Construct star_wars_matrix
box_office <- c(460.998, 314.4, 290.475, 247.900, 309.306, 165.8)
region <- c("US", "non-US")
```

```
titles <- c("A New Hope",
                 "The Empire Strikes Back",
                 "Return of the Jedi")

star_wars_matrix <- matrix(box_office,
                      nrow = 3, byrow = TRUE,
                      dimnames = list(titles, region))

# Calculate worldwide box office figures
worldwide_vector <- rowSums(star_wars_matrix)
all_wars_matrix <- cbind(star_wars_matrix,worldwide_vector)
# star_wars_matrix and star_wars_matrix2 are available in your workspace
star_wars_matrix
```

```
##                              US non-US
## A New Hope              460.998  314.4
## The Empire Strikes Back 290.475  247.9
## Return of the Jedi      309.306  165.8
```

```
# Assign to the variable theory what this chapter is about!
theory <- "factors"
# Sex vector
sex_vector <- c("Male", "Female", "Female", "Male", "Male")

# Convert sex_vector to a factor
factor_sex_vector <- factor(sex_vector)

# Print out factor_sex_vector
print(factor_sex_vector )
```

```
## [1] Male   Female Female Male   Male
## Levels: Female Male
```

```
# Animals
animals_vector <- c("Elephant", "Giraffe", "Donkey", "Horse")
factor_animals_vector <- factor(animals_vector)
factor_animals_vector
```

```
## [1] Elephant Giraffe  Donkey   Horse
## Levels: Donkey Elephant Giraffe Horse
```

```
# Temperature
temperature_vector <- c("High", "Low", "High","Low", "Medium")
factor_temperature_vector <- factor(temperature_vector, order = TRUE, levels = c("Low", "Medium", "High
factor_temperature_vector
```

```
## [1] High   Low    High   Low    Medium
## Levels: Low < Medium < High
```

the order with which you assign the levels is important. If you type levels(factor_survey_vector), you'll see that it outputs [1] "F" "M". If you don't specify the levels of the factor when creating the vector, R will automatically assign them alphabetically. To correctly map "F" to "Female" and "M" to "Male", the levels should be set to c("Female", "Male"), in this order.

```
# Code to build factor_survey_vector
survey_vector <- c("M", "F", "F", "M", "M")
factor_survey_vector <- factor(survey_vector)
# Specify the levels of factor_survey_vector
levels(factor_survey_vector) <- c("Female", "Male")
factor_survey_vector
```

```
## [1] Male   Female Female Male   Male  
## Levels: Female Male
```

```
# Build factor_survey_vector with clean levels
survey_vector <- c("M", "F", "F", "M", "M")
factor_survey_vector <- factor(survey_vector)
levels(factor_survey_vector) <- c("Female", "Male")
factor_survey_vector
```

```
## [1] Male   Female Female Male   Male  
## Levels: Female Male
```

```
# Generate summary for survey_vector
summary(survey_vector)
```

```
##    Length     Class      Mode 
##         5 character character
```

```
# Generate summary for factor_survey_vector
summary(factor_survey_vector)
```

```
## Female   Male 
##      2      3
```

```
# Build factor_survey_vector with clean levels
survey_vector <- c("M", "F", "F", "M", "M")
factor_survey_vector <- factor(survey_vector)
levels(factor_survey_vector) <- c("Female", "Male")

# Male
male <- factor_survey_vector[1]

# Female
female <- factor_survey_vector[2]

# Battle of the sexes: Male 'larger' than female?
male > female
```

```
## Warning in Ops.factor(male, female): ’>’ not meaningful for factors
```

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

```
# Create speed_vector
speed_vector <- c("medium", "slow", "slow", "medium", "fast")
# Convert speed_vector to ordered factor vector
factor_speed_vector <- factor(speed_vector, ordered = TRUE, levels = c("slow", "medium", "fast"))
# Print factor_speed_vector
factor_speed_vector
```

```
## [1] medium slow   slow   medium fast  
## Levels: slow < medium < fast
```

```
summary(factor_speed_vector)
```

```
##   slow medium   fast 
##      2      2      1
```

```
# Investigate the structure of mtcars
head(mtcars)
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
str(mtcars)
```

```
## ’data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

```
# Definition of vectors
name <- c("Mercury", "Venus", "Earth",
          "Mars", "Jupiter", "Saturn",
          "Uranus", "Neptune")
type <- c("Terrestrial planet",
          "Terrestrial planet",
          "Terrestrial planet",
          "Terrestrial planet", "Gas giant",
          "Gas giant", "Gas giant", "Gas giant")
diameter <- c(0.382, 0.949, 1, 0.532,
              11.209, 9.449, 4.007, 3.883)
rotation <- c(58.64, -243.02, 1, 1.03,
              0.41, 0.43, -0.72, 0.67)
rings <- c(FALSE, FALSE, FALSE, FALSE, TRUE, TRUE, TRUE, TRUE)

# Create a data frame from the vectors
planets_df <- data.frame(name, type, diameter, rotation, rings)
```

```
# Check the structure of planets_df
str(planets_df)
```

```
## ’data.frame’:    8 obs. of  5 variables:
##  $ name    : chr  "Mercury" "Venus" "Earth" "Mars" ...
##  $ type    : chr  "Terrestrial planet" "Terrestrial planet" "Terrestrial planet" "Terrestrial planet
##  $ diameter: num  0.382 0.949 1 0.532 11.209 ...
##  $ rotation: num  58.64 -243.02 1 1.03 0.41 ...
##  $ rings   : logi  FALSE FALSE FALSE FALSE TRUE TRUE ...
```

```
# The planets_df data frame from the previous exercise is pre-loaded
head(planets_df)
```

```
##      name               type diameter rotation rings
## 1 Mercury Terrestrial planet    0.382    58.64 FALSE
## 2   Venus Terrestrial planet    0.949  -243.02 FALSE
## 3   Earth Terrestrial planet    1.000     1.00 FALSE
## 4    Mars Terrestrial planet    0.532     1.03 FALSE
## 5 Jupiter          Gas giant   11.209     0.41  TRUE
## 6  Saturn          Gas giant    9.449     0.43  TRUE
```

```
# Print out diameter of Mercury (row 1, column 3)
planets_df[1,3]
```

```
## [1] 0.382
```

```
# Print out data for Mars (entire fourth row)
planets_df[4, ]
```

```
##   name               type diameter rotation rings
## 4 Mars Terrestrial planet    0.532     1.03 FALSE
```

```
# The planets_df data frame from the previous exercise is pre-loaded
head(planets_df)
```

```
##      name               type diameter rotation rings
## 1 Mercury Terrestrial planet    0.382    58.64 FALSE
## 2   Venus Terrestrial planet    0.949  -243.02 FALSE
## 3   Earth Terrestrial planet    1.000     1.00 FALSE
## 4    Mars Terrestrial planet    0.532     1.03 FALSE
## 5 Jupiter          Gas giant   11.209     0.41  TRUE
## 6  Saturn          Gas giant    9.449     0.43  TRUE
```

```r
# Select first 5 values of diameter column
planets_df[1:5, "diameter"]
```

```
## [1]  0.382  0.949  1.000  0.532 11.209
```

```r
# Select the rings variable from planets_df
rings_vector <- planets_df$rings

# Print out rings_vector
rings_vector
```

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

```r
# Adapt the code to select all columns for planets with rings
planets_df[rings_vector, ]
```

```
##      name      type diameter rotation rings
## 5 Jupiter Gas giant   11.209     0.41  TRUE
## 6  Saturn Gas giant    9.449     0.43  TRUE
## 7  Uranus Gas giant    4.007    -0.72  TRUE
## 8 Neptune Gas giant    3.883     0.67  TRUE
```

```r
# Select planets with diameter < 1
subset(planets_df, subset = diameter < 1)
```

```
##      name               type diameter rotation rings
## 1 Mercury Terrestrial planet    0.382    58.64 FALSE
## 2   Venus Terrestrial planet    0.949  -243.02 FALSE
## 4    Mars Terrestrial planet    0.532     1.03 FALSE
```

```r
# Use order() to create positions
positions <- order(planets_df$diameter)
# Use positions to sort planets_df
planets_df[positions, ]
```

```
##      name               type diameter rotation rings
## 1 Mercury Terrestrial planet    0.382    58.64 FALSE
## 4    Mars Terrestrial planet    0.532     1.03 FALSE
## 2   Venus Terrestrial planet    0.949  -243.02 FALSE
## 3   Earth Terrestrial planet    1.000     1.00 FALSE
## 8 Neptune          Gas giant    3.883     0.67  TRUE
## 7  Uranus          Gas giant    4.007    -0.72  TRUE
## 6  Saturn          Gas giant    9.449     0.43  TRUE
## 5 Jupiter          Gas giant   11.209     0.41  TRUE
```

```r
# Vector with numerics from 1 up to 10
my_vector <- 1:10
# Matrix with numerics from 1 up to 9
my_matrix <- matrix(1:9, ncol = 3)
# First 10 elements of the built-in data frame mtcars
```

```
my_df <- mtcars[1:10,]
# Construct list with these different elements:
my_list <- list(my_vector, my_matrix, my_df)
my_list
```

```
## [[1]]
##  [1]  1  2  3  4  5  6  7  8  9 10
## 
## [[2]]
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
## 
## [[3]]
##                    mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
```

```
# Adapt list() call to give the components names
my_list <- list(vec = my_vector, mat = my_matrix, df = my_df)
# Print out my_list
my_list
```

```
## $vec
##  [1]  1  2  3  4  5  6  7  8  9 10
## 
## $mat
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
## 
## $df
##                    mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
```