Assignment\_01\_NoskyChristopher.R

Chris

2020-12-06

# Assignment: ASSIGNMENT 1  
# Name: Lastname, Firstname  
# Date: 2010-02-14  
  
## Create a numeric vector with the values of 3, 2, 1 using the `c()` function  
## Assign the value to a variable named `num\_vector`  
## Print the vector  
  
num\_vector <- c(3, 2, 1)  
num\_vector

## [1] 3 2 1

## Create a character vector with the values of "three", "two", "one" "using the `c()` function  
## Assign the value to a variable named `char\_vector`  
## Print the vector  
  
char\_vector <- c('three', 'two', 'one')  
char\_vector

## [1] "three" "two" "one"

## Create a vector called `week1\_sleep` representing how many hours slept each night of the week  
## Use the values 6.1, 8.8, 7.7, 6.4, 6.2, 6.9, 6.6  
  
week1\_sleep <- c("Sunday" = 6.1, "Monday" = 8.8, "Tuesday" = 7.7,   
 "Wednesday" = 6.4, "Thursday" = 6.2, "Friday" = 6.9,  
 "Saturday" = 6.6)  
week1\_sleep

## Sunday Monday Tuesday Wednesday Thursday Friday Saturday   
## 6.1 8.8 7.7 6.4 6.2 6.9 6.6

## Display the amount of sleep on Tuesday of week 1 by selecting the variable index  
  
week1\_sleep[3]

## Tuesday   
## 7.7

## Create a vector called `week1\_sleep\_weekdays`  
## Assign the weekday values using indice slicing  
  
week1\_sleep\_weekdays <- week1\_sleep[1:7]  
week1\_sleep\_weekdays

## Sunday Monday Tuesday Wednesday Thursday Friday Saturday   
## 6.1 8.8 7.7 6.4 6.2 6.9 6.6

## Add the total hours slept in week one using the `sum` function  
## Assign the value to variable `total\_sleep\_week1`  
  
total\_sleep\_week1 <- sum(week1\_sleep)  
total\_sleep\_week1

## [1] 48.7

## Create a vector called `week2\_sleep` representing how many hours slept each night of the week  
## Use the values 7.1, 7.4, 7.9, 6.5, 8.1, 8.2, 8.9  
week2\_sleep <- c('Sunday' = 7.1, 'Monday' = 7.4, 'Tuesday' = 7.9,  
 'Wednesday' = 6.5, 'Thursday' = 8.1, 'Friday' = 8.2,  
 'Saturday' = 8.9)  
week2\_sleep

## Sunday Monday Tuesday Wednesday Thursday Friday Saturday   
## 7.1 7.4 7.9 6.5 8.1 8.2 8.9

## Add the total hours slept in week two using the sum function  
## Assign the value to variable `total\_sleep\_week2`  
total\_sleep\_week2 <- sum(week2\_sleep)  
total\_sleep\_week2

## [1] 54.1

## Determine if the total sleep in week 1 is less than week 2 by using the < operator  
  
total\_sleep\_week1 < total\_sleep\_week2

## [1] TRUE

## Calculate the mean hours slept in week 1 using the `mean()` function  
  
mean(week1\_sleep\_weekdays)

## [1] 6.957143

## Create a vector called `days` containing the days of the week.  
## Start with Sunday and end with Saturday  
days <- c('Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday')  
  
## Assign the names of each day to `week1\_sleep` and `week2\_sleep` using the `names` function and `days` vector  
names(week1\_sleep) <- days  
names(week2\_sleep) <- days  
week1\_sleep

## Sunday Monday Tuesday Wednesday Thursday Friday Saturday   
## 6.1 8.8 7.7 6.4 6.2 6.9 6.6

## Display the amount of sleep on Tuesday of week 1 by selecting the variable name  
week1\_sleep['Tuesday']

## Tuesday   
## 7.7

## Create vector called weekdays from the days vector  
weekdays <- days[2:6]  
weekdays

## [1] "Monday" "Tuesday" "Wednesday" "Thursday" "Friday"

## Create vector called weekends containing Sunday and Saturday  
weekends <- days[-(2:6)]  
weekends

## [1] "Sunday" "Saturday"

## Calculate the mean about sleep on weekdays for each week  
## Assign the values to weekdays1\_mean and weekdays2\_mean  
  
weekdays1\_mean <- mean(week1\_sleep[weekdays])  
weekdays2\_mean <- mean(week2\_sleep[weekdays])  
weekdays1\_mean

## [1] 7.2

weekdays2\_mean

## [1] 7.62

## Using the weekdays1\_mean and weekdays2\_mean variables,  
## see if weekdays1\_mean is greater than weekdays2\_mean using the `>` operator  
weekdays1\_mean > weekdays2\_mean

## [1] FALSE

## Determine how many days in week 1 had over 8 hours of sleep using the `>` operator  
week1\_sleep[weekdays] > 8

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

## Create a matrix from the following three vectors  
  
student01 <- c(100.0, 87.1)  
student02 <- c(77.2, 88.9)  
student03 <- c(66.3, 87.9)  
  
students\_combined <- c(student01, student02, student03)  
grades <- matrix(c(students\_combined), nrow = 3, byrow = TRUE)  
  
grades

## [,1] [,2]  
## [1,] 100.0 87.1  
## [2,] 77.2 88.9  
## [3,] 66.3 87.9

## Add a new student row with `rbind()`  
student04 <- c(95.2, 94.1)  
grades <- rbind(grades, student04)  
grades

## [,1] [,2]  
## 100.0 87.1  
## 77.2 88.9  
## 66.3 87.9  
## student04 95.2 94.1

## Add a new assignment column with `cbind()`  
assignment04 <- c(92.1, 84.3, 75.1, 97.8)  
grades <- cbind(grades, assignment04)  
grades

## assignment04  
## 100.0 87.1 92.1  
## 77.2 88.9 84.3  
## 66.3 87.9 75.1  
## student04 95.2 94.1 97.8

## Add the following names to columns and rows using `rownames()` and `colnames()`  
assignments <- c("Assignment 1", "Assignment 2", "Assignment 3")  
students <- c("Florinda Baird", "Jinny Foss", "Lou Purvis", "Nola Maloney")  
  
rownames(grades) <- students  
colnames(grades) <- assignments  
grades

## Assignment 1 Assignment 2 Assignment 3  
## Florinda Baird 100.0 87.1 92.1  
## Jinny Foss 77.2 88.9 84.3  
## Lou Purvis 66.3 87.9 75.1  
## Nola Maloney 95.2 94.1 97.8

## Total points for each assignment using `colSums()`  
total\_points <- colSums(grades)  
total\_points

## Assignment 1 Assignment 2 Assignment 3   
## 338.7 358.0 349.3

## Total points for each student using `rowSums()`  
total\_points\_student <- rowSums(grades)  
total\_points\_student

## Florinda Baird Jinny Foss Lou Purvis Nola Maloney   
## 279.2 250.4 229.3 287.1

## Matrix with 10% and add it to grades  
weighted\_grades <- grades \* 0.1 + grades  
weighted\_grades

## Assignment 1 Assignment 2 Assignment 3  
## Florinda Baird 110.00 95.81 101.31  
## Jinny Foss 84.92 97.79 92.73  
## Lou Purvis 72.93 96.69 82.61  
## Nola Maloney 104.72 103.51 107.58

## Create a factor of book genres using the genres\_vector  
## Assign the factor vector to factor\_genre\_vector  
genres\_vector <- c("Fantasy", "Sci-Fi", "Sci-Fi", "Mystery", "Sci-Fi", "Fantasy")  
factor\_genre\_vector <- genres\_vector  
factor\_genre\_vector

## [1] "Fantasy" "Sci-Fi" "Sci-Fi" "Mystery" "Sci-Fi" "Fantasy"

## Use the `summary()` function to print a summary of `factor\_genre\_vector`  
summary(factor\_genre\_vector)

## Length Class Mode   
## 6 character character

## Create ordered factor of book recommendations using the recommendations\_vector  
## `no` is the lowest and `yes` is the highest  
recommendations\_vector <- c("neutral", "no", "no", "neutral", "yes")  
factor\_recommendations\_vector <- factor(  
 recommendations\_vector,  
 ordered = TRUE,  
 levels = c('no', 'neutral', 'yes')  
)  
factor\_recommendations\_vector

## [1] neutral no no neutral yes   
## Levels: no < neutral < yes

## Use the `summary()` function to print a summary of `factor\_recommendations\_vector`  
summary(factor\_recommendations\_vector)

## no neutral yes   
## 2 2 1

## Using the built-in `mtcars` dataset, view the first few rows using the `head()` function  
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

## Using the built-in mtcars dataset, view the last few rows using the `tail()` function  
tail(mtcars)

## mpg cyl disp hp drat wt qsec vs am gear carb  
## Porsche 914-2 26.0 4 120.3 91 4.43 2.140 16.7 0 1 5 2  
## Lotus Europa 30.4 4 95.1 113 3.77 1.513 16.9 1 1 5 2  
## Ford Pantera L 15.8 8 351.0 264 4.22 3.170 14.5 0 1 5 4  
## Ferrari Dino 19.7 6 145.0 175 3.62 2.770 15.5 0 1 5 6  
## Maserati Bora 15.0 8 301.0 335 3.54 3.570 14.6 0 1 5 8  
## Volvo 142E 21.4 4 121.0 109 4.11 2.780 18.6 1 1 4 2

## Create a dataframe called characters\_df using the following information from LOTR  
name <- c("Aragon", "Bilbo", "Frodo", "Galadriel", "Sam", "Gandalf", "Legolas", "Sauron", "Gollum")  
race <- c("Men", "Hobbit", "Hobbit", "Elf", "Hobbit", "Maia", "Elf", "Maia", "Hobbit")  
in\_fellowship <- c(TRUE, FALSE, TRUE, FALSE, TRUE, TRUE, TRUE, FALSE, FALSE)  
ring\_bearer <- c(FALSE, TRUE, TRUE, FALSE, TRUE, TRUE, FALSE, TRUE, TRUE)  
age <- c(88, 129, 51, 7000, 36, 2019, 2931, 7052, 589)  
  
characters\_df <- data.frame(name, race, in\_fellowship, ring\_bearer, age)  
characters\_df

## name race in\_fellowship ring\_bearer age  
## 1 Aragon Men TRUE FALSE 88  
## 2 Bilbo Hobbit FALSE TRUE 129  
## 3 Frodo Hobbit TRUE TRUE 51  
## 4 Galadriel Elf FALSE FALSE 7000  
## 5 Sam Hobbit TRUE TRUE 36  
## 6 Gandalf Maia TRUE TRUE 2019  
## 7 Legolas Elf TRUE FALSE 2931  
## 8 Sauron Maia FALSE TRUE 7052  
## 9 Gollum Hobbit FALSE TRUE 589

## Sorting the characters\_df by age using the order function and assign the result to the sorted\_characters\_df  
sorted\_characters\_df <- characters\_df[order(age),]  
sorted\_characters\_df

## name race in\_fellowship ring\_bearer age  
## 5 Sam Hobbit TRUE TRUE 36  
## 3 Frodo Hobbit TRUE TRUE 51  
## 1 Aragon Men TRUE FALSE 88  
## 2 Bilbo Hobbit FALSE TRUE 129  
## 9 Gollum Hobbit FALSE TRUE 589  
## 6 Gandalf Maia TRUE TRUE 2019  
## 7 Legolas Elf TRUE FALSE 2931  
## 4 Galadriel Elf FALSE FALSE 7000  
## 8 Sauron Maia FALSE TRUE 7052

## Use `head()` to output the first few rows of `sorted\_characters\_df`  
head(sorted\_characters\_df)

## name race in\_fellowship ring\_bearer age  
## 5 Sam Hobbit TRUE TRUE 36  
## 3 Frodo Hobbit TRUE TRUE 51  
## 1 Aragon Men TRUE FALSE 88  
## 2 Bilbo Hobbit FALSE TRUE 129  
## 9 Gollum Hobbit FALSE TRUE 589  
## 6 Gandalf Maia TRUE TRUE 2019

## Select all of the ring bearers from the dataframe and assign it to ringbearers\_df  
ringbearers\_df <- characters\_df[characters\_df$ring\_bearer == TRUE,]  
ringbearers\_df

## name race in\_fellowship ring\_bearer age  
## 2 Bilbo Hobbit FALSE TRUE 129  
## 3 Frodo Hobbit TRUE TRUE 51  
## 5 Sam Hobbit TRUE TRUE 36  
## 6 Gandalf Maia TRUE TRUE 2019  
## 8 Sauron Maia FALSE TRUE 7052  
## 9 Gollum Hobbit FALSE TRUE 589

## Use `head()` to output the first few rows of `ringbearers\_df`  
head(ringbearers\_df)

## name race in\_fellowship ring\_bearer age  
## 2 Bilbo Hobbit FALSE TRUE 129  
## 3 Frodo Hobbit TRUE TRUE 51  
## 5 Sam Hobbit TRUE TRUE 36  
## 6 Gandalf Maia TRUE TRUE 2019  
## 8 Sauron Maia FALSE TRUE 7052  
## 9 Gollum Hobbit FALSE TRUE 589