DSC520_Week2_Guruprasad_Velikadu_Assignment01

Guruprasad Velikadu Krishnamoorthy

2022-12-11

Contents

1 Assignment 01

1.1 Create a numeric vector with the values of 3, 2, 1 using the `c()` function;

1.1.1 Assign the value to a variable named `num_vector`

1.1.2 Print the vector

```
num_vector <- c(3,2,1)
num_vector
```

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

1.2 Create a character vector with the values of “three”, “two”, “one” “using the `c()` function

1.2.1 Assign the value to a variable named `char_vector`

1.2.2 Print the vector

```
char_vector <- c("three","two","one")
char_vector
```

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

1.3 Create a vector called `week1_sleep` representing how many hours slept each night of the week

1.3.1 Use the values 6.1, 8.8, 7.7, 6.4, 6.2, 6.9, 6.6

```
week1_sleep <- c(6.1, 8.8, 7.7, 6.4, 6.2, 6.9, 6.6)
```

1.4 Display the amount of sleep on Tuesday of week 1 by selecting the variable index

```
week1_sleep[3]
```

```
## [1] 7.7
```

1.5 Create a vector called `week1_sleep_weekdays`

1.5.1 Assign the weekday values using indice slicing

```
week1_sleep_weekdays <- week1_sleep[2:6]
```

1.6 Add the total hours slept in week one using the `sum` function

1.6.1 Assign the value to variable `total_sleep_week1`

```
total_sleep_week1 <- sum(week1_sleep)
```

1.7 Create a vector called `week2_sleep` representing how many hours slept each night of the week

1.7.1 Use the values 7.1, 7.4, 7.9, 6.5, 8.1, 8.2, 8.9

```
week2_sleep <- c(7.1, 7.4, 7.9, 6.5, 8.1, 8.2, 8.9)
```

1.8 Add the total hours slept in week two using the sum function

1.8.1 Assign the value to variable `total_sleep_week2`

```
total_sleep_week2 <- sum(week2_sleep)
```

1.9 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
```

1.10 Calculate the mean hours slept in week 1 using the `mean()` function

```
mean(week1_sleep)
```

```
## [1] 6.957143
```

1.11 Create a vector called `days` containing the days of the week.

1.11.1 Start with Sunday and end with Saturday

```
days <- c("Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday")
```

1.12 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
```

1.13 Display the amount of sleep on Tuesday of week 1 by selecting the variable name

```
week1_sleep[3]
```

```
## Tuesday
##     7.7
```

1.14 Create vector called weekdays from the days vector

```
weekdays <- days[2:6]
```

1.15 Create vector called weekends containing Sunday and Saturday

```
weekends <- days[c(1,7)]
```

1.16 Calculate the mean about sleep on weekdays for each week

1.16.1 Assign the values to weekdays1_mean and weekdays2_mean

```
weekdays1_mean <- mean(week1_sleep[weekdays])
weekdays2_mean <- mean(week2_sleep[weekdays])
```

1.17 Using the weekdays1_mean and weekdays2_mean variables,

1.17.1 see if weekdays1_mean is greater than weekdays2_mean using the > operator

```
weekdays1_mean > weekdays2_mean
```

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

1.18 Determine how many days in week 1 had over 8 hours of sleep using the > operator

```
week1_sleep > 8
```

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

1.19 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(students_combined, byrow = TRUE, nrow = 3)
```

1.20 Add a new student row with `rbind()`

```
student04 <- c(95.2, 94.1)
grades <- rbind(grades,student04)
```

1.21 Add a new assignment column with `cbind()`

```r
assignment04 <- c(92.1, 84.3, 75.1, 97.8)
grades <- cbind(grades,assignment04)
```

1.22 Add the following names to columns and rows using `rownames()` and `colnames()`

```r
assignments <- c("Assignment 1", "Assignment 2", "Assignment 3")
students <- c("Florinda Baird", "Jinny Foss", "Lou Purvis", "Nola Maloney")
rownames(grades) <- students
colnames(grades) <- assignments
```

1.23 Total points for each assignment using `colSums()`

```r
colSums(grades)
```

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

1.24 Total points for each student using `rowSums()`

```r
rowSums(grades)
```

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

1.25 Matrix with 10% and add it to grades

```r
weighted_grades <- grades * 0.1 + grades
```

1.26 Create a factor of book genres using the genres_vector

1.26.1 Assign the factor vector to factor_genre_vector

```r
genres_vector <- c("Fantasy", "Sci-Fi", "Sci-Fi", "Mystery", "Sci-Fi", "Fantasy")
factor_genre_vector <- as.factor(genres_vector)
```

1.27 Use the `summary()` function to print a summary of `factor_genre_vector`

```r
summary(factor_genre_vector)
```

```
## Fantasy Mystery  Sci-Fi
##       2       1       3
```

1.28 Create ordered factor of book recommendations using the recommendations_vector

1.28.1 `no` is the lowest and `yes` is the highest

```r
recommendations_vector <- c("neutral", "no", "no", "neutral", "yes")
factor_recommendations_vector <- factor(
  recommendations_vector,
  ordered = TRUE,
  levels = c("no","neutral","yes")
)
```

1.29 Use the `summary()` function to print a summary of `factor_recommendations_vector`

```r
summary(factor_recommendations_vector)
```

```
##      no neutral     yes
##       2       2       1
```

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

```r
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
```

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

```r
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
```

1.32 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)
```

1.33 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),]
```

1.34 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
```

1.35 Select all of the ring bearers from the dataframe and assign it to ringbearers_df

```
ringbearers_df <- characters_df[characters_df$ring_bearer == TRUE,]
```

1.36 Use `head()` to output the first few rows of `ringbearers_df`

```r
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
```

2 Session Info

```r
sessionInfo()
```

```
## R version 4.2.2 (2022-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 22621)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## loaded via a namespace (and not attached):
##  [1] compiler_4.2.2  magrittr_2.0.3  fastmap_1.1.0   cli_3.4.1
##  [5] tools_4.2.2     htmltools_0.5.3 yaml_2.3.6      stringi_1.7.8
##  [9] rmarkdown_2.18  knitr_1.41      stringr_1.4.1   xfun_0.34
## [13] digest_0.6.30   rlang_1.0.6     evaluate_0.18
```