

# assignment\_01\_KummarikuntaVidyasagar.R

12702

2020-09-04

```
# Assignment: ASSIGNMENT 1
# Name: Kummarikunta, Vidyasagar
# Date: 2020-09-04

## Create a numeric vector with the values of 3, 2, 1 using the 'c()' function
## Assign the value to a variable named 'num_vector'
#num_vector <- __
num_vector <- c(3,2,1)

## Print the vector
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'
#char_vector <- __

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

## Print the vector
char_vector

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

## 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 <- __

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

#print week1_sleep week
week1_sleep

## [1] 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_sleepp[_]
week1_sleep[3]

## [1] 7.7

## Create a vector called 'week1_sleep_weekdays'
## Assign the weekday values using indice slicing
```

```

#week1_sleep_weekdays <- week1_sleep[__:__]

week1_sleep_weekdays <- week1_sleep[2:6]

#print week1_sleep_weekdays

week1_sleep_weekdays

## [1] 8.8 7.7 6.4 6.2 6.9

## Add the total hours slept in week one using the 'sum' function
## Assign the value to variable 'total_sleep_week1'
#total_sleep_week1 <- ___
total_sleep_week1 <- sum(week1_sleep)

#Print total_sleep_week1

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 <- ___

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

#Print week2_sleep values

week2_sleep

## [1] 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 <- ___

total_sleep_week2 <- sum(week2_sleep)

#Print total_sleep_week2

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(__)
mean(week1_sleep)

## [1] 6.957143

```

```

## Create a vector called 'days' containing the days of the week.
## Start with Sunday and end with Saturday
#days <- ___

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

#Print days

days

## [1] "Sunday"      "Monday"      "Tuesday"     "Wednesday"  "Thursday"   "Friday"
## [7] "Saturday"

## Assign the names of each day to 'week1_sleep' and 'week2_sleep' using the 'names' function and 'days'
#names(week1_sleep) <- ___
#names(week2_sleep) <- ___

names(week1_sleep) <- days
names(week2_sleep) <- days

## Display the amount of sleep on Tuesday of week 1 by selecting the variable name
#week1_sleep[___]

week1_sleep["Tuesday"]

## Tuesday
##      7.7

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

weekdays <- days[2:6]

#Print weekdays

weekdays

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

## Create vector called weekends containing Sunday and Saturday
#weekends <- ___

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

#Print weekends

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])

## 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 > 8

##      Sunday      Monday      Tuesday Wednesday Thursday      Friday      Saturday
##      FALSE       TRUE       FALSE      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 <- __

students_combined <- c(student01, student02, student03)

students_combined

## [1] 100.0  87.1  77.2  88.9  66.3  87.9

#grades <- matrix(students_combined, byrow = __, nrow = __)

grades <- matrix(students_combined, byrow = TRUE, nrow = 3)

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)

# print grades to check student04 is added or not
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 <- cbind(grades, assignment04)

```

```

## 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(__) <- __
#colnames(__) <- __

rownames(grades) <- students
colnames(grades) <- assignments

## Total points for each assignment using 'colSums()'
#__

colSums(grades)

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

## Total points for each student using 'rowSums()'
#__

rowSums(grades)

## 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

## 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 <- ___
factor_genre_vector <- genres_vector

## Use the 'summary()' function to print a summary of 'factor_genre_vector'
#summary(__)

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 = __,
  ordered = TRUE,
  #levels = c(__, __, __)
  levels = c("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(__, __, __, __, __)
```

```
characters_df <- data.frame(name, race, in_fellowship, ring_bearer, age)
```

```
## 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(__),]
sorted_characters_df <- characters_df[order(age),]
```

```
## 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$__ == __,]
```

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

```
## 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
```