The results below are generated from an R script.

```
# Assignment: ASSIGNMENT 1
# Name: TANG, XIN
# Date: 2023-06-12

## 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)

## 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")

## 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(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[2]

## [1] 8.8

## Create a vector called `week1_sleep_weekdays`
## Assign the weekday values using indice slicing
week1_sleep_weekdays <- week1_sleep[1:5]

## 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)

## 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(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)

## 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)

## [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
```

```r
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["Tuesday"]

## Tuesday
##     7.7

## Create vector called weekdays from the days vector
weekdays <- days[2:7]

## Create vector called weekends containing Sunday and Saturday
weekends <- c("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 <- cbind(student01, student02, student03)
grades <- matrix(students_combined, byrow = 2, nrow = 3)

## Add a new student row with `rbind()`
student04 <- c(95.2, 94.1)
grades <- rbind(student01, student02, student03, student04)

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

```r
## 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 <- genres_vector

## 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 = is.ordered(recommendations_vector),
  levels = c("no", "neutral", "yes")
)
print(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, n=3)

##                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

## Using the built-in mtcars dataset, view the last few rows using the `tail()` function
tail(mtcars, n=3)
```

```
##                mpg cyl disp  hp drat   wt qsec vs am gear carb
## Ferrari Dino  19.7   6  145 175 3.62 2.77 15.5  0  1    5    6
## Maserati Bora 15.0   8  301 335 3.54 3.57 14.6  0  1    5    8
## Volvo 142E    21.4   4  121 109 4.11 2.78 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)
print(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_charact
sorted_characters_df <- characters_df[order(age),]
## Use `head()` to output the first few rows of `sorted_characters_df`
head(sorted_characters_df, n=3)

##     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

## Select all of the ring bearers from the dataframe and assign it to ringbearers_df
ringbearers_df <- characters_df[characters_df$ring_bearer == 'TRUE',]
## Use `head()` to output the first few rows of `ringbearers_df`
head(ringbearers_df, n=3)

##    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
```

The R session information (including the OS info, R version and all packages used):

```
sessionInfo()

## R version 4.3.0 (2023-04-21 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19045)
##
## Matrix products: default
```

```
##
##
## locale:
## [1] LC_COLLATE=English_United States.utf8  LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8 LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: America/Chicago
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] tinytex_0.45 knitr_1.43
##
## loaded via a namespace (and not attached):
## [1] compiler_4.3.0 tools_4.3.0    highr_0.10     xfun_0.39      evaluate_0.21
```

```
Sys.time()
```

```
## [1] "2023-06-13 20:52:59 CDT"
```