Class 06: R Functions Lab

Selma Cifric (PID A69042976)

Table of contents

Q1

> Q1. Write a function grade() to determine an overall grade from a vector of student homework assignment scores dropping the lowest single score. If a student misses a homework (i.e. has an NA value) this can be used as a score to be potentially dropped. Your final function should be adquately explained with code comments and be able to work on an example class gradebook such as this one in CSV format: "https://tinyurl.com/gradeinput"

```
#First I will access the url and call that data set grades

url <- "https://tinyurl.com/gradeinput"
grades<- read.csv(url)
grades
```

```
           X hw1 hw2 hw3 hw4 hw5
1  student-1 100  73 100  88  79
2  student-2  85  64  78  89  78
3  student-3  83  69  77 100  77
4  student-4  88  NA  73 100  76
5  student-5  88 100  75  86  79
6  student-6  89  78 100  89  77
```

```
7   student-7  89 100  74  87 100
8   student-8  89 100  76  86 100
9   student-9  86 100  77  88  77
10 student-10  89  72  79  NA  76
11 student-11  82  66  78  84 100
12 student-12 100  70  75  92 100
13 student-13  89 100  76 100  80
14 student-14  85 100  77  89  76
15 student-15  85  65  76  89  NA
16 student-16  92 100  74  89  77
17 student-17  88  63 100  86  78
18 student-18  91  NA 100  87 100
19 student-19  91  68  75  86  79
20 student-20  91  68  76  88  76
```

```r
# My goal for Q1: find the lowest score (or NA) then drop it, calculate the
# average of scores and then turn it into a grade. First, function `grade()` is a
# function that takes in students' scores. Using function `min()` will find the
# smallest numeric score. It will find its position in the vector
# [which(scores == min(...))] and take the first time smallest score occurs and
# label it as NA (... <- NA). na.rm=TRUE means that the function will ignore
# already existing NAs.

grade <- function(scores) {
  scores[which(scores == min(scores, na.rm = TRUE))[1]] <- NA


# To find an average score after dropping the lowest score, while ignoring NAs,
# we use simple `mean()` function:

  avg <- mean(scores, na.rm = TRUE)

# Now that we have dropped the lowest score and averaged the rest, we can assign
# letter grades with `if()` and `else()` loops. In case the student did not have
# any numeric scores, just NAs, in that case our average is missing and final
# grade letter for the student is NA. To check if the average is missing we can
# use `is.na(avg)` and if true, then print "NA".

   letter <- if (is.na(avg)) NA else
    if (avg >= 90) "A" else
    if (avg >= 80) "B" else
    if (avg >= 70) "C" else
    if (avg >= 60) "D" else "F"

# Now, to combine average score (after dropping the lowest one) which is found
# in avg vector, with letter grade assigned for that average score, found in
# letter grade object, we make a new `return()` vector. It will give us an
# average score and letter grade for each student.

return(c(average_score = avg, grade = letter)) }

# Finally, I want to apply the function of grade on my data set so that every
# student has an average score and letter grade. Function `t(apply()` will bring
# back student names that were dropped before with `grades[, -1]` function.

grades_final <- cbind(
  grades[, 1, drop = FALSE],
  t(apply(grades[, -1], 1, grade))
)

grades_final
```

```
            X    average_score grade
1   student-1            91.75     A
2   student-2             82.5     B
3   student-3            84.25     B
4   student-4               88     B
5   student-5            88.25     B
6   student-6               89     B
7   student-7               94     A
8   student-8            93.75     A
9   student-9            87.75     B
10 student-10 81.3333333333333     B
11 student-11               86     B
12 student-12            91.75     A
13 student-13            92.25     A
14 student-14            87.75     B
15 student-15 83.3333333333333     B
16 student-16             89.5     B
17 student-17               88     B
18 student-18               97     A
19 student-19            82.75     B
20 student-20            82.75     B
```

Q2

> Q2. Using your grade() function and the supplied gradebook, Who is the top scoring student overall in the gradebook?

First I make sure that the final average grade is a number with `str()` function.

```
str(grades_final)
```

```
'data.frame':   20 obs. of  3 variables:
 $ X            : chr  "student-1" "student-2" "student-3" "student-4" ...
 $ average_score: chr  "91.75" "82.5" "84.25" "88" ...
 $ grade        : chr  "A" "B" "B" "B" ...
```

This output tells me my average is a character, which is not good. I will convert chr to num first to be sure. To find a top student, I will combine `which()` and `max()` functions on average column in my final grades data set.

```
grades_final$average_score <- as.numeric(grades_final$average_score)

top_student <- which.max(grades_final$average)
cat("Top student is", top_student)
```

```
Top student is 18
```

Q3

> Q3. From your analysis of the gradebook, which homework was toughest on students (i.e. obtained the lowest scores overall?)

For this question, we are going back to the original data set. My goal is to average each hw column, ignoring NA scores, and find `which()` hw has `min()` score. But first, I need to drop the first column "x" because it contains non-numeric values, which gives me an error.

```
hw_score <- grades[, -1]
hw_score_average <- colMeans(hw_score, na.rm=TRUE)
hw_score_average
```

```
     hw1      hw2      hw3      hw4      hw5 
89.00000 80.88889 80.80000 89.63158 83.42105 
```

Now that I have an average of each column, I will pick the minimum score average.

```
toughest_hw <- hw_score_average[which.min(hw_score_average)]
toughest_hw
```

```
 hw3
80.8
```