Class 06: R Functions Lab

AUTHOR
Hyeseung (Frankie) Son PID: A16025601

Hints

Once you have a working function for vector inputs (such as the student1, student2, and student3 vectors below) you can use the apply() function to work with data frame inputs such as those obtained from read.csv(). Additional functions you will want to explore include mean(), is.na(), which.min(), which.max(), sum(), and cor(). Remember, you can ask for help on any function by typing a question mark before the function name e.g. ?sum.

```
# Example input vectors to start with
student1<-c(100, 100, 100, 100, 100, 100, 100, 90)
student2<-c(100, NA, 90, 90, 90, 90, 97, 80)
student3<-c(90, NA, NA, NA, NA, NA, NA, NA)
```

> 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 adequately 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” [3pts]

We can use `mean()` function to calculate the average score for one student.

```
mean(student1)
```

```
[1] 98.75
```

What about student 3? Use the argument na.rm=TRUE to remove NA values from the student scores and calculate the mean.

```
mean(student3, na.rm=TRUE)
```

```
[1] 90
```

Use the function is.na() to identify NA values and their position in the vector.

```
student2
```

```
[1] 100  NA  90  90  90  90  97  80
```

```
is.na(student2)
```

```
[1] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
which(is.na(student2))
```

```
[1] 2
```

We can also replace the missed NA values with a score of 0. I can also make these values anything I want (zero, TRUE, etc)

```
student2[is.na(student2)] <- 0
student2
```

```
[1] 100   0  90  90  90  90  97  80
```

I will use temp object (`x`) so I don't screw up my original objects.

```
x<-student2
x[is.na(x)] <- 0
mean(x)
```

```
[1] 79.625
```

We want to drop the lowest score before calculating the mean. Each student will drop their worst assignment score. Use the which.min(x) function to locate which position the lowest assignment score is.

```
x<-student1
x[which.min(x)]
```

```
[1] 90
```

```
x[8]
```

```
[1] 90
```

As you can see, x[which.min(x)] and x[8] both give us the minimum scores for student 1. I can use the minus sign with `which.min()` to display all scores excluding the lowest value.

```
x[-which.min(x)]
```

```
[1] 100 100 100 100 100 100 100
```

Now I just need to put all this back together to make our working snippet code:

```
x<- student2
#Map/replace NA values for zero
x[is.na(x)] <- 0
```

```r
# Eclude the lowest value and cslculate the mean
mean( x[-which.min(x) ] )
```

```
[1] 91
```

This is my working snippet that I can turn into a function called `grade()`.

All functions in R have at least 3 things: - **Name**, in our case “grade” - **Input arguments** - **Body**, this is our working snippet from above.

```r
grade <- function(x) {
#Map/replace NA values for zero
x[is.na(x)] <- 0

# Eclude the lowest value and cslculate the mean
mean( x[-which.min(x) ] )
}
```

Can I use this function now? Make sure you run the code to let the R brain know of this new `function()` by running the chunk before using the function.

```r
grade(student1)
```

```
[1] 100
```

Now let’s read a gradebook from online:

```r
hw <- read.csv("https://tinyurl.com/gradeinput", row.names = 1)
hw
```

```
           hw1 hw2 hw3 hw4 hw5
student-1  100  73 100  88  79
student-2   85  64  78  89  78
student-3   83  69  77 100  77
student-4   88  NA  73 100  76
student-5   88 100  75  86  79
student-6   89  78 100  89  77
student-7   89 100  74  87 100
student-8   89 100  76  86 100
student-9   86 100  77  88  77
student-10  89  72  79  NA  76
student-11  82  66  78  84 100
student-12 100  70  75  92 100
student-13  89 100  76 100  80
student-14  85 100  77  89  76
student-15  85  65  76  89  NA
student-16  92 100  74  89  77
student-17  88  63 100  86  78
student-18  91  NA 100  87 100
```

```
student-19  91  68  75  86  79
student-20  91  68  76  88  76
```

Use the `apply()` function to grade all students in the class with our new `grade()` function.

The `apply()` function allows us to run over any function with the rows and columns of a data.frame. We will use rows because we want to calculate the highest overall grade by student (row = 1).

The general format of the apply layer is: apply(name of data variable, margin=1 for rows OR margin=2 for columns, function)

```
apply(hw, 1, grade)
```

```
 student-1  student-2  student-3  student-4  student-5  student-6  student-7 
     91.75      82.50      84.25      84.25      88.25      89.00      94.00 
 student-8  student-9 student-10 student-11 student-12 student-13 student-14 
     93.75      87.75      79.00      86.00      91.75      92.25      87.75 
student-15 student-16 student-17 student-18 student-19 student-20 
     78.75      89.50      88.00      94.50      82.75      82.75 
```

> Q2. Using your grade() function and the supplied gradebook, Who is the top scoring student overall in the gradebook? [3pts]

We will assign a new variable ans to the `apply()` layer we created to find the overall score for each student. We will also use `which,max` to find the student with the overall top score.

```
ans <- apply(hw, 1, grade)
ans[which.max(ans)]
```

```
student-18 
      94.5 
```

> Q3. From your analysis of the gradebook, which homework was toughest on students (i.e. obtained the lowest scores overall? [2pts]

We are still looking through the data varaible `hw` but we are now looking for the lowest scoring homework , which means we will be looking at the margin `columns= 2`. We'll use the `mean` function to calculate the mean across all rows.

Wrapping this entire snippet uder the `which.min()` will give us which homework scores

```
avg.scores <- apply(hw, 2, mean, na.rm = TRUE)
which.min(avg.scores)
```

```
hw3 
  3 
```

```
tot.scores <- apply(hw, 2, sum, na.rm = TRUE)
which.min(tot.scores)
```

```
hw2
  2
```

```
avg.scores
```

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

```
tot.scores
```

```
 hw1  hw2  hw3  hw4  hw5
1780 1456 1616 1703 1585
```

Homework 2 and 3 were the lowest scoring homeworks, based either on average scores or total sum of scores.

> Q4. Optional Extension: From your analysis of the gradebook, which homework was most predictive of overall score (i.e. highest correlation with average grade score)? [1pt]

```
ans
```

```
 student-1  student-2  student-3  student-4  student-5  student-6  student-7
     91.75      82.50      84.25      84.25      88.25      89.00      94.00
 student-8  student-9 student-10 student-11 student-12 student-13 student-14
     93.75      87.75      79.00      86.00      91.75      92.25      87.75
student-15 student-16 student-17 student-18 student-19 student-20
     78.75      89.50      88.00      94.50      82.75      82.75
```

We will use the `cor()` fucntion to generate the correlation value for each assignment. A correlation value closer to 1, means

```
cor(hw$hw1, ans)
```

```
[1] 0.4250204
```

```
cor(hw$hw3, ans)
```

```
[1] 0.3042561
```

If I try ot get hw2 correlation values, I get NA as there are missing homeworks.

```
cor(hw$hw2, ans)
```

```
[1] NA
```

I will mask all NA values in all assignments to zero.

```
mask <- hw
mask[is.na(mask)] <- 0
```

We see that homework 5 has relatively a high correlation value, meaning the hw5 score was highly predictive of a student's overall scores.

```
cor(mask$hw5, ans)
```

```
[1] 0.6325982
```

Can we find the correlation values for all asignments with one fucntion?

```
apply(mask, 2, cor, y=ans)
```

```
      hw1       hw2       hw3       hw4       hw5 
0.4250204 0.1767780 0.3042561 0.3810884 0.6325982 
```

So this function generates the correlation values for all assignments. And let's just display the maximum value.

```
which.max(apply(mask, 2, cor, y=ans))
```

```
hw5 
  5 
```

We confirm that hw5 has the highest correlation value of 0.6325982, meaning the hw5 score was most highly predictive of a student's overall scores.

> Q5. Make sure you save your Quarto document and can click the "Render" (or Rmarkdown"Knit") button to generate a PDF foramt report without errors. Finally, submit your PDF to gradescope. [1pt]