Class06: R Functions

Brian Wells (A69026838)

All about functions in R

Everyfunction in R has at least 3 things: - name(you pick it) - arguments (the input(s) of your function), - the body

Today we will write a function to grade a class of student assignment scores (e.g. homeworks, etc).

First I'll work with a simplified vector input where I know what the answer should be.

```
# 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 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"

```
student1
```

```
[1] 100 100 100 100 100 100 100  90
```

use `mean()` to get the average for each student

```
mean(student1)
```

```
[1] 98.75
```

to find vector of smallest score use `which.min()`

```r
which.min(student1)
```

```
[1] 8
```

to see what value the score is put `student#[#]`

```r
student1[8]
```

```
[1] 90
```

to get get the min value via the position vector

```r
student1[which.min(student1)]
```

```
[1] 90
```

to get everything *but* min value

```r
student1[-which.min(student1)]
```

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

to get the students average of everything but lowest score

```r
mean(student1[-which.min(student1)])
```

```
[1] 100
```

```r
#this can be used in the function
```

now for student 2 with an NA

```r
student2
```

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

function removes NA. this is a bad idea.

```r
mean(student2, na.rm=T)
```

```
[1] 91
```

we need to make NA=0

```r
x <- student2
x
```

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

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

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

```r
mean(x)
```

```
[1] 79.625
```

now to combine the parts that worked!

```r
x <- student3
x[is.na(x)] <- 0
mean(x[-which.min(x)])
```

```
[1] 12.85714
```

Function Time!!

combine our snippets into a function

```r
grade <- function(x){
  x[is.na(x)] <- 0
  mean(x[-which.min(x)])
}
```

let's test our function

```r
grade(student1)
```

```
[1] 100
```

```r
grade(student2)
```

```
[1] 91
```

```r
grade(student3)
```

```
[1] 12.85714
```

Now let’s apply this function to a full class.

```r
# now grade all students in an example class
url <- "https://tinyurl.com/gradeinput"
gradebook <- read.csv(url,
                      row.names = 1)
```

here is the function from before with some code notes

```r
is.na(gradebook)
```

```
             hw1   hw2   hw3   hw4   hw5
student-1  FALSE FALSE FALSE FALSE FALSE
student-2  FALSE FALSE FALSE FALSE FALSE
student-3  FALSE FALSE FALSE FALSE FALSE
student-4  FALSE  TRUE FALSE FALSE FALSE
student-5  FALSE FALSE FALSE FALSE FALSE
student-6  FALSE FALSE FALSE FALSE FALSE
student-7  FALSE FALSE FALSE FALSE FALSE
student-8  FALSE FALSE FALSE FALSE FALSE
student-9  FALSE FALSE FALSE FALSE FALSE
student-10 FALSE FALSE FALSE  TRUE FALSE
student-11 FALSE FALSE FALSE FALSE FALSE
student-12 FALSE FALSE FALSE FALSE FALSE
student-13 FALSE FALSE FALSE FALSE FALSE
student-14 FALSE FALSE FALSE FALSE FALSE
```

```
student-15 FALSE FALSE FALSE FALSE  TRUE
student-16 FALSE FALSE FALSE FALSE FALSE
student-17 FALSE FALSE FALSE FALSE FALSE
student-18 FALSE  TRUE FALSE FALSE FALSE
student-19 FALSE FALSE FALSE FALSE FALSE
student-20 FALSE FALSE FALSE FALSE FALSE
```

```
grade <- function(x){
  ##assign NA to a value of 0 -- missing scores get a score of 0
  x[is.na(x)] <- 0
  ##average while dropping the lowest score
  mean(x[-which.min(x)])
}
```

now let's use `apply()` to grade the entire gradebook!

```
apply(gradebook,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?

```
#let's get our gradebook results
results <- apply(gradebook,1,grade)
#sort them from highest to lowest score
sort(results, decreasing=T)
```

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

```r
#return the student with the highest value for their score
which.max(results)
```

```
student-18
        18
```

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

First we want to get an average of the different columns (bc the columns are the different assignments)

lets interpret using average

```r
## use 2 to get columns
apply(gradebook,2,mean)
```

```
 hw1  hw2  hw3  hw4  hw5 
89.0   NA 80.8   NA   NA 
```

oops, gave us NA on any homeworks that have them if we think that the NA should be counted as a 0 and the most 0s correlates to most difficult–

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

```
           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   0  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   0  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   0
student-16  92 100  74  89  77
student-17  88  63 100  86  78
student-18  91   0 100  87 100
student-19  91  68  75  86  79
student-20  91  68  76  88  76
```

```r
which.min(apply(mask,2,mean))
```

```
hw2
  2
```

if you think NA should be removed (not counted as a 0) because skipping doesn't necessarily mean "difficult" –

```r
hw.ave<- apply(gradebook,2,mean,na.rm=T)
which.min(hw.ave)
```

```
hw3
  3
```

> 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]

lets get correlation for just one homework

```r
gradebook[is.na(gradebook)] <- 0
cor(results,gradebook$hw2)
```

```
[1] 0.176778
```

Now lets get the correlation for all the homeworks

```r
hw.cor <- apply(gradebook,2,cor,x=results)
##now print the max cor value of all the homeworks
which.max(hw.cor)
```

```
hw5
  5
```

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.