R Functions Lab

Rogelio Castro

Student Grades:

```
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)
```

```
mean(student1)
```

```
[1] 98.75
```

We can use Mean to obtain the average of Student 1.

```
mean(student2, na.rm=TRUE)
```

```
[1] 91
```

If we decide to use the same function `mean()` for Student 2 we get an error because there is a non-numeric value in NA. `na.rm()` will remove the NA an the output will be the average of the student without the NA value.

What about student 3?

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

```
[1] 90
```

If we use the same code as Student 2 `mean(x, na.rm=TRUE)` we will not have a fair approach to grading since Student 3 only did 1 assignment worth 90%, and the others are missing. `na.rm=TRUE` is removing the missing assignments with NA values and only averaging the 90% assignment. Giving them a 90% for one assignment in all of the class.

So how do we approach this? Find where the NA values are Using `is.na()` function might help

```
student2
```

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

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

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

`is.na()` is helping us to identify where the NA value is.

```
student2[ is.na(student2) ]
```

```
[1] NA
```

`student2[is.na(student2)]` is helping us know if there is/are NA values and how many (spelled out)

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

```
[1] 2
```

`which(is.na(student2))` is helping us know if the numerical value of the NA values available

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

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

It is time to work with new temp object (that I will call `x`) so I don't screw up my original objects.

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

```
[1] 11.25
```

Finally we wan to drop the lowest score before calculating the mean. This is equivalent to allowing the students to drop their worst assignment score.

```
x <- student1
x
```

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

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

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

Now we put everything together to make the working snippet:

```
x<- student3
x
```

```
[1] 90 NA NA NA NA NA NA NA
```

```
# Map/Replace NA values to zero:
x[ is.na(x)] <- 0

#Exclude the lowest score and calculate the mean:
mean( x[ -which.min(x) ] )
```

```
[1] 12.85714
```

Cool! this works. Now let's turn it into a function called `grade()`

All functions in R have at least 3 things:

- **Name**, in our case "grade"
- Input **arguments**, student 1 etc.
- **Body**, this is our working snippet above

```r
grade<- function(x) {

# Map/Replace NA values to zero:
x[ is.na(x)] <- 0

#Exclude the lowest score and calculate the mean:
mean( x[ -which.min(x) ] )
}
```

Can I use the function now? Make sure to press the play button to let the machine know about function `grade(x)`

```r
grade(student1)
```

```
[1] 100
```

Read 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
```

We can use the `apply()` function to grade all the students in this class with out new `grade()` function.

The `apply()` functions allows us to run any function over with the rows or columns of a data.frame. Let's see how it works:

```
ans <- apply(hw, 1, grade)
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 
```

What we did was `apply(data (hw), margin (1 for rows and 2 for columns, function (grade) )`

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

```
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]

```
apply(hw, 2, mean, na.rm=TRUE)
```

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

```
which.min( apply(hw, 2, mean, na.rm=TRUE) )
```

```
hw3 
  3 
```

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

```
hw3
  3
```

```
apply(hw, 2, sum, na.rm=TRUE)
```

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

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

```
hw2
  2
```

```
tot.scores
```

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

```
ave.scores
```

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

> 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]

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

```
[1] 0.4250204
```

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

```
[1] 0.3042561
```

If I try on hw2 I get Na as there are missing homeworks (i.e. NA values)

```
hw$hw2
```

```
 [1]  73  64  69  NA 100  78 100 100 100  72  66  70 100 100  65 100  63  NA  68
[20]  68
```

I will nask all NA values to zero.

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

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

```
[1] 0.6325982
```

We can use the `apply()` function here on the columns of hw (i.e. the individual homeworks) and pass it the overall scores for the class (in my `ans` object as an extra argument)

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

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