Class 06: R Functions

Lance Boling

```
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)
```

```
 (sum(student1)- min(student1))/7
```

```
[1] 100
```

Or

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

```
[1] 100
```

to exclude NA values from Student2:

```
mean(student2[-which.min(student2)], na.rm=TRUE)
```

```
[1] 92.83333
```

to assign Student to as "x"

```
x <- student2
x
```

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

ChatGPT: "To convert NA (missing) values to zero in R, you can use the is.na() function to identify the missing values and then use logical indexing to replace them with zeros."

```
x <- student3
x[is.na(x)] <- 0
mean(x[-which.min(x)], na.rm=TRUE)
```

```
[1] 12.85714
```

Now x can be changed to whichever student we want to grade.

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" [3pts]

```
grade <- function(x) {
  #convert/Mask NA values to zero
  x[is.na(x)] <- 0
  #drop lowest score and get the mean. Note na.rm=TRUE is not required since NA was assign
  mean(x[-which.min(x)])
}
```

To read the gradebook and convert first row to names:

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

```
           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
```

To Use the apply command to perform a batch function on all grades:

```
apply(gradebook, MARGIN=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
```

```
#Note: instead of MARGIN=1 could just use a "1".  2 would average columns and 3 averages r
```

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

```
which.max(apply(gradebook, MARGIN=1, grade))
```

```
student-18
        18
```

Could also assign the “answer” to ans so we can easily query the results

```
ans <- apply(gradebook, MARGIN=1, grade)
which.max(ans)
```

```
student-18
        18
```

Q3. From your analysis of the gradebook, which homework was toughest on students (i.e. obtained the lowest scores overall? [2pts] –>change the margin to columns (2). However this still drops the lowest score which skews the result

```r
which.min(apply(gradebook, MARGIN=2, grade))
```

```
hw2
  2
```

instead:

```r
mask <- gradebook
mask[is.na(mask)] <- 0
hw.ave <- (apply(mask, 2, mean))
which.min(hw.ave)
```

```
hw2
  2
```

We could also sum the columns and then choose the lowest to determine the lowest scoring quiz:

```r
gradebook[is.na(gradebook)] <- 0
which.min(apply (gradebook, 2, sum))
```

```
hw2
  2
```

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]

```r
cor(mask$hw4, ans)
```

```
[1] 0.3810884
```

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

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

```
#then can find which is the most correlated with which.max
which.max(apply(mask, 2, cor, y=ans))
```

```
hw5
  5
```