Class_6

Create a function that averages the students score with the lowest grade dropped.

Anagha Pashilkar A16450123

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"

```
grade <- function(student) {
   #returns True or False if value is or is not "na"
   is.na(student)

   #indexes values which are "na" and replaces with 0
   student[which(is.na(student))] <- 0

   #average of all values with lowest value dropped
   mean(student[-which.min(student)])
 }
```

```
url <- "https://tinyurl.com/gradeinput"
   gradebook <- read.csv(url, row.names = 1)
   #reads csv file and saves data as gradebook.
   #argues for row one and after to be read (so student name isn't incuded as a row of dat

 #apply(x, margin, function)
   #x = an array
   #margin = 1 is row and 2 is column... we want to look at avg of each row
   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? [**3pts**]

run apply function and **save results** (save as variable)

```r
results <- apply(gradebook, 1, grade)

#sorts from largest to smallest value.
#sort([data set], arguement)

sort(results, decreasing = TRUE)
```

```
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 
```

Find top scoring student

```r
which.max(results)
```

```
student-18 
        18 
```

Q3.Find toughest homework

```r
hw_avg <- apply(gradebook, 2, mean, na.rm=TRUE)
which.min(hw_avg)
```

```
hw3 
  3 
```

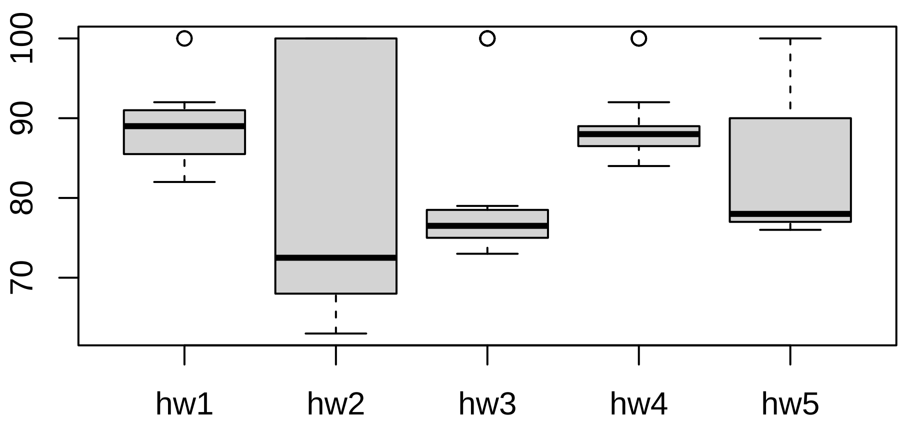
```
#toughest hw based on lowest hw avg
```

```
med.scores <- apply(gradebook, 2, median, na.rm=TRUE)
which.min(med.scores)
```

```
hw2
  2
```

```
#toughest hw based on lowest median out of the 4
```

```
boxplot(gradebook)
```


hw1
hw2
hw3
hw4
hw5