Class 06: R functions

Barry (PID: 911)

All about functions in R

Every function in R has at least 3 things: - name (you pick it) - arguments (the input(s) to your function), and - the body.

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

First I will 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)
```

Let's start slow and find the average for student1

```
mean(student1)
```

```
[1] 98.75
```

How can we drop the lowest score? I can use the `min()` function to find the lowest score (element in the vector).

```
min(student1)
```

```
[1] 90
```

I found the function `which.min()` let's try it out...

```
student1
```

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

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

```
[1] 8
```

```
student1[ -8 ]
```

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

Let's put the use of `which.min()`, minus indexing and `mean()` together to solve this baby.

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

```
[1] 100
```

Will this work for student2?

```
mean( student2[ -which.min(student2) ])
```

```
[1] NA
```

```
x <- student3
mean( x[ -which.min(x) ])
```

```
[1] NA
```

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

```
[1] 90
```

```
student3
```

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

We can "mask" the NA or change them to be zero. The rational here is if you don't do a hw you get zero pts.

We can use the `is.na()` function to find where the missing homeworks are in the input vector.

```
x <- student2
is.na(x)
```

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

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

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

I think we are there. Let's put these pices together to solve this.

```
x <- student3
# Mask NA to zero
x[ is.na(x) ] <- 0
# Find the mean dropping the lowest score
mean( x[ -which.min(x) ])
```

```
[1] 12.85714
```

Turn this snippet into a function.

```
grade <- function(x) {
  # Mask NA to zero
  x[ is.na(x) ] <- 0
  # Find the mean dropping the lowest score
  mean( x[ -which.min(x) ])
}
```

We can use this function now to grade any student

```
grade(student1)
```

```
[1] 100
```

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]

I need to read the gradebook CSV file

```
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
```

A very useful function that Barry is forcing us to use here is the `apply()` function. How do we use it to take our new `grade()` function and apply it over the full gradebook.

```
ans <- apply(gradebook, 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
```

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

```
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]

We are going to use the `apply()` function again here…

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

```
hw3
  3
```

Let's mask the NA values to zero

```
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
```

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

```
hw2
  2
```

```
which.min( apply(mask, 2, sum) )
```

```
hw2
  2
```

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

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

```
[1] 0.176778
```

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

```
[1] 0.6325982
```

Now take the `apply()` function and the `cor()` function and run over our whole gradebook.

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

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

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

```
hw5
  5
```