Class 6 BIMM 143

Mariam Benny (A17103451)

Today we are going to explore R functions and begin to think about writing our own functions

Let's start simple and write our first function to add some numbers

Every function in R has three things - a **name**, we pick this - one or more input or **arguments** - the **body**, where the work gets done

```
add <- function(x, y=1, z=0) {
  x + y + z
}
```

Now let's try it out

```
add(x=c(10, 1, 1, 10), y=1)
```

```
[1] 11  2  2 11
```

```
add(10)
```

```
[1] 11
```

```
add(10,10)
```

```
[1] 20
```

```
add(10,10,20)
```

```
[1] 40
```

```
mean(c(10,10,NA), na.rm=T)
```

```
[1] 10
```

#Lab sheet work

```
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)
```

Begin by calculating the avg for student1

```
mean(student1)
```

```
[1] 98.75
```

Try student2

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

```
[1] 91
```

Try student3

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

```
[1] 90
```

We also want to drop the lowest score

```
student1
```

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

```
student1[-8]
```

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

```
mean(student1[-8])
```

```
[1] 100
```

We can try the `min()` function to find lowest score

```
min(student1)
```

```
[1] 90
```

I want to find the location of the min vale, not the value itself, for this I can use `which.min()`

```
student1
```

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

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

```
[1] 8
```

Let's put these two together

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

```
[1] 100
```

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

```
[1] 100
```

```
x <- student2
x
```

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

```
x[2] <- 0
x
```

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

```
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
```

So far we have a working snippet:

```
x <- student2
#Find NAs in 'x' and make them zero
x[ is.na(x) ] <- 0

#Finds the min value and removes it before getting mean
mean( x[-which.min(x)] )
```

```
[1] 91
```

> 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]

Now turn it into a function

```
grade <- function(x) {
  #Find NAs in 'x' and make them zero
x[ is.na(x) ] <- 0

#Finds the min value and removes it before getting mean
mean( x[-which.min(x)] )
}
```

```
grade(student1)
```

```
[1] 100
```

```
grade(student2)
```

```
[1] 91
```

```
grade(student3)
```

```
[1] 12.85714
```

Now `apply()` to our class gradebook

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

To use the `apply()` function on this `gradebook` dataset, I need to decide whether I want to "apply" the `grade()` function over the rows (1) or columns (2) of the `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? [3pts]

Using the results from the code below, the top scoring student is student 18 with a 94.5% grade.

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

```
student-18
        18
```

```
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]

The homework that was the toughest on students was homework 2.

```
q3 <- apply(gradebook, 2, grade)
which.min(q3)
```

```
hw2
  2
```

```
q3[which.min(q3)]
```

```
     hw2
76.63158
```

```
masked_gradebook <- gradebook
masked_gradebook [is.na(masked_gradebook)] = 0
apply(masked_gradebook, 2, mean)
```

```
  hw1   hw2   hw3   hw4   hw5
89.00 72.80 80.80 85.15 79.25
```

```
grade2 <- function(x, drop.low=TRUE) {
  #Find NAs in 'x' and make them zero
  x[ is.na(x) ] <- 0

  if(drop.low) {
  cat("Hello low")

#Drops lowest value and find mean
  out <- mean( x[-which.min(x)] )
  } else {
    out <- mean(x)
    cat("No low")
  }
  return(out)
}
```

```
grade2(student1, FALSE)
```

```
No low
```

```
[1] 98.75
```

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]

Using the code below, hw5 has the highest correlation with 0.63.

```
apply(masked_gradebook, 2, cor, ans)
```

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

Extra code: The function to calculate correlation in R is called `cor()`

```
x <- c(100, 90, 80, 100)
y <- c(100, 90, 80, 100)
z <- c(80, 90, 100, 10)

cor(x, y)
```

```
[1] 1
```

```
cor(x, z)
```

```
[1] -0.6822423
```

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

```
[1] 0.4250204
```

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

```
[1] 0.6325982
```

I want to `apply()` the `car()` function over the `masked_gradebook()` and use the `ans` scores for the class

```
apply(masked_gradebook, 2, cor, ans)
```

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