Class 6: R functions

Bianca Barriga

Functions in R

In this class we will work through the process of developing our own function for calculating average grades for fictional students in a fictional class.

We are going to start with a simplified version of the problem where I know what the answer should be.

Example input vectors to start with

```
# 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)
```

To get the average grade we can use the function `mean()`.

```
mean(student1)
```

```
[1] 98.75
```

The `min()` will return the smallest value.

```
min(student1)
```

```
[1] 90
```

and related function `which.min()` will return the index of the min value

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

```
[1] 8
```

```
x <- 1:5
y <- which.max(x)
z <- (x[-y])
mean(z)
```

```
[1] 2.5
```

```
x[-3]
```

```
[1] 1 2 4 5
```

Let's put these bits together to find the average score dropping the lowest single score.

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

```
[1] 100
```

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

```
[1] NA
```

We need to remove NA's from student 2 gradebook.

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

```
[1] 92.83333
```

```
student3
```

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

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

```
[1] NaN
```

We need to find NA and replace with 0. Then calculate the mean.

```
y <- replace(student3, is.na(student3), 0)
mean(y[-which.min(y)])
```

```
[1] 12.85714
```

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

```
[1] 12.85714
```

We now have our working code snipet that can become the body of our function.

Recall that all functions in R have at least 3 things: -name(we pick this) -arguments(input to the function) body(where the work gets done)

```
grades <- function(y) {
  z <- replace(y, is.na(y), 0)
  mean(z[-which.min(z)])}
```

```
grades(student1)
```

```
[1] 100
```

```
grades(student2)
```

```
[1] 91
```

```
grades(student3)
```

```
[1] 12.85714
```

Establish a new function.

```
grade <- function(x) {
  #map NA values to zero
  x[is.na(x)] <- 0
  mean(x[-which.min(x)])
}
```

Lers use this new function `grade()`

```
grade(student1)
```

```
[1] 100
```

```
grade(student2)
```

```
[1] 91
```

```
grade(student3)
```

```
[1] 12.85714
```

> 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]

To read this CSV file we are going to use the `read.csv()`.

```
gradebook <- read.csv("https://tinyurl.com/gradeinput")
```

As is, the student names are under data points and not official row names, so we need to modify the inputs to specify row names.

```
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
```

Now use our function for this gradebook example

This wont work.

```
#grade(gradebook)
```

We can use the `apply()` function to grade all the students in this gradbook. The `apply()` function will apply any function over the rows (MARGIN = 1) or columns (MARGIN=2) of any data.frame/matrix etc, #apply function inputs, 1 indicates rows, 2 indicates columns

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

```
 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(results)
```

```
student-18
        18
```

Q3. From your analysis of the gradebook, which homework was toughest on students (i.e. obtained the lowest scores overall? [2pts]

```
which.min(apply(gradebook,2,sum,na.rm = T))
```

```
hw2
  2
```

I guess we need to mask those 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,sum,na.rm = T))
```

```
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]

```
#results
mask$hw5
```

```
 [1]  79  78  77  76  79  77 100 100  77  76 100 100  80  76   0  77  78 100  79
[20]  76
```

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

```
[1] 0.6325982
```

```
#results
cor(mask$hw2, results)
```

```
[1] 0.176778
```

Can we use the `apply()` function to do this all for us?

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

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