Class 6: R functions

Cynthia Perez (A16393492)

Functions are how we get work done in R. We call functions to do everything from reading data to doing analysis and outputing plots and results.

All functions in R have at least three things:

- a **name** (you pick this)
- input **arguments** (there can be one or many)
- the **body** (where the work gets done, this is the code between the curly brackets)

A first function

Let's write a function to add some numbers. We can call it `add()`

```
x <- 10
y <- 10
x + y
```

```
[1] 20
```

```
add <- function(x) {
  y =10
  x + y
}
```

Can I just use my new function?

```
add(1)
```

```
[1] 11
```

Let's make it a bit more flexible.

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

add(10,10)
```

```
[1] 20
```

```r
add(10)
```

```
[1] 11
```

2nd example grade() function

Write a function to grade student work.

We will start simple with a version of the problem and the following example student vectors:

```r
# 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)
```

Start with student1

```r
mean(student1)
```

```
[1] 98.75
```

Mean of student2

```r
mean(student2, na.rm=TRUE)
```

```
[1] 91
```

Mean of student3

```r
mean(student3, na.rm=TRUE)
```

```
[1] 90
```

Okay lets try to work with student1 and find(and drop) the lowest score.

```
student1
```

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

Use the `min()` and `which.min()` functions to find lowest score:

```
min(student1)
```

```
[1] 90
```

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

```
[1] 8
```

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

```
[1] 90
```

Remaining values for student1 without lowest score:

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

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

Take the mean of student1 excluding the lowest score:

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

```
[1] 100
```

Shorten the code

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

```
[1] 100
```

Our Approach to NA problem (missing homework) we can replace NA values with 0

First task is to find the NA values (i.e., where are they in the vector)

```
x <- student2
x
```

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

```
is.na(x)
```

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

I have found the NA (TRUE) values from `is.na()` now I want to make them equal to zero.

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

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

I want to combine the `is.na(x)` and take this "masked" (vector of student scores with NA values as zero) and drop the lowest to get the mean.

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

```
[1] 91
```

Now onto student3 with the working code snippet.

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

```
[1] 12.85714
```

Now we can turn this snippet into my first function.

```
grade <- function(x) {
  # Make NA values equal to zero
  x[is.na(x)] <- 0
  # Drop lowest score and take the mean
  mean(x[-which.min(x)])
}
```

Test run the function on students 1, 2, and 3

```
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]

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

The `apply()` function in R is helpful but can be a little confusing to begin with. Use apply function to use the grade function on all students in the gradebook.

```
#use 1 for rows and 2 for columns
class <- apply(gradebook, 1, grade)
class
```

```
 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(class)
```

```
student-18
        18
```

```
max(class)
```

```
[1] 94.5
```

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

Use `which.min()` function and change the margin argument for `apply()` to 2 (gives means of the columns-homeworks).

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

```
hw3
  3
```

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]

use `cor()` function to determine correlation between homeworks and grades for the entire class.

```
#class
cor(gradebook$hw5, class)
```

```
[1] NA
```

```
gradebook$hw5
```

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

```
#have NAs in some assignments
```

Need to mask all NAs to zero

```
mask <- gradebook
mask[is.na(mask)] <- 0
```

Now we can take the correlation with NAs set to zero.

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

```
[1] 0.6325982
```

Now we can use `apply()` to examine the correlation of every assignment in the masked gradebook to the overall score for the class.

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

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

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

```
hw5
  5
```