Class 6: R functions

Gaeun: A16814573

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 at least 3 things:

- a **name**, we pick this
- one or more input **arguments**
- the **body**, where the work gets done

```
# y has a default value of 1, z has a default value of 0
add <- function(x, y=1, z=0) {
  x + y + z
}
```

Now let's try it out.

```
add(10, 1)
```

```
[1] 11
```

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

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

```
add(10)
```

```
[1] 11
```

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

```
[1] 20
```

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

```
[1] 40
```

```r
# na.rm overrides the NA
mean( c(10, 10, NA), na.rm=TRUE)
```

```
[1] 10
```

Lab sheet work

> 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"

```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)
```

Begin by caluclating the average for student 1

```r
student1
```

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

```r
mean(student1)
```

```
[1] 98.75
```

Try on student 2

```r
student2
```

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

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

```
[1] 91
```

Try on student 3

```r
student3
```

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

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

```
[1] 90
```

Hm…. this sucks! I need to try something else and come back to this issue of missing values (NAs).

We also want to drop the lowest score from a given student's set of scores.

```r
student1
```

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

```r
# this removes the 8th value
student1[-8]
```

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

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

```r
min(student1)
```

```
[1] 90
```

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

```r
student1
```

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

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

```
[1] 8
```

Let's put these two things together.

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

```
[1] 8
```

```r
student1[-8]
```

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

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

```
[1] 100
```

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

```
[1] 100
```

```r
# or
mean(student1[-which.min(student1)])
```

```
[1] 100
```

Now trying on student 2. But we need to deal with NA (missing values) somehow..

One idea is we make all the NA values zero.

```r
x <- student2
x
```

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

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

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

```r
x[is.na(x)]
```

```
[1] NA
```

```r
x[!is.na(x)]
```

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

So far we have a working snippet.

```r
# Find NAs in `x` and makes them 0
x[is.na(x)] <- 0

# Drops lowest value and finds mean
mean(x[-which.min(x)])
```

```
[1] 91
```

Now turn it into a function

```r
grade <- function(x) {
  x[is.na(x)] <- 0
  mean(x[-which.min(x)])
}
```

```r
grade(student1)
```

```
[1] 100
```

```r
grade(student2)
```

```
[1] 91
```

```r
grade(student3)
```

```
[1] 12.85714
```

But instead of calling each student, we want the function to grade all the students at once.

Now `apply()` to our class gradebook

```r
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
```

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()`.

```r
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?

```r
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)?

First use the `apply` function and change the margin to 2 so that it can indicate the columns (homework).

```r
hw <- apply(gradebook, 2, grade)
hw
```

```
     hw1      hw2      hw3      hw4      hw5 
89.36842 76.63158 81.21053 89.63158 83.42105 
```

THe problem is that it doesn't take into account that that NA homeworks are counted as 0.

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

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

Now find the homework score that obtained the lowest score.

```r
which.min(hw)
```

```
hw2
  2
```

I could modify the `grade()` function to this too - i.e. not drop the lowest options

```r
grade2 <- function(x, drop.low=TRUE) {

  # Finds NAs in `x` and makes them 0
  x[is.na(x)] <- 0

  if(drop.low) {
    cat("Hello low")
    # Drop 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)?

The function to calculate correlations in R is called `cor()`

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

```r
# 0 means no correlation at all, 1 means perfectly correlated, -1 means perfectly anti-correl
cor(x,y)
```

```
[1] 1
```

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

```
[1] -0.6822423
```

```r
cor(ans, masked_gradebook$hw1)
```

```
[1] 0.4250204
```

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

```r
predict <- apply(masked_gradebook, 2, cor, y=ans)
```

Find the highest homework correlation.

```r
which.max(predict)
```

```
hw5
  5
```