Class 6: R Functions

Chloe Do

Function basics

All functions in R consist of at least 3 things:

- A **name** (we can pick this but it must start with a character)
- Input **arguments** (there can be multiple comm separated inputs)
- The **body** (where work actually happens)

```
# 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)
```

I can start by using the `mean()` function to calculate an average.

```
mean(student1)
```

```
[1] 98.75
```

I found the `min()` function to find the minimum valye in a vector.

```
min(student1)
```

```
[1] 90
```

Looking at the "See Also" section of the `min()` help page. I found out about `which.min()`

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

```
[1] 8
```

```
  student1
```

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

```
  student1[1:7]
```

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

I can get the same vector without the 8th element with the minus index trick…

```
  student1[-8]
```

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

So I will combine the output of `which.min()` with the minus index trick to get the student score without the lowest value

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

```
[1] 100
```

Hmm… For student2 this gives NA

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

```
[1] NA
```

I see there is an `na.rm=FALSE` by default to the `mean()` function. Will this help us?

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

```
[1] 92.83333
```

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

```
[1] NA
```

Well that sucks! We need another way …

How about we replace all NA (missing values) with zero.

```
  student3
```

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

```
  is.na(student3)
```

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

```
  student3[is.na(student3)] <- 0
  student3
```

```
[1] 90  0  0  0  0  0  0  0
```

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

```
[1] 12.85714
```

All this copy paste is silly and dangerous - time to write a function

```
  x <- student1
  x[is.na(x)] <- 0
  mean(x[-which.min(x)])
```

```
[1] 100
```

I now have my working snippet of code that I have simplified to work with any student `x`.

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

```
[1] 100
```

Now turn into a function:

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

```
grade(student1)
```

```
[1] 100
```

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

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

Have a look at the first 6 rows

```
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
```

Time to learn about the `apply()` function.

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

Which student did the best overall?

```
which.max(results)
```

```
student-18
        18
```

```
results[which.max(results)]
```

```
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]

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

```
hw2
  2
```

```
lowest_hw <- apply(gradebook, 2, grade)
```

Which homework is the toughest?

```
which.min(lowest_hw)
```

```
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]

```
mask <- gradebook
mask[is.na(mask)] <- 0
cor(mask$hw5, results)
```

```
[1] 0.6325982
```

```
cor(mask$hw1, results)
```

```
[1] 0.4250204
```

Or use apply…

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

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