Class 06: R Functions

Marina Puffer (PID:A16341339)

All about functions in R

Functions are the way we get stuff done in R. We call a function to read data, compute stuff, plot stuff, etc.

R makes writing functions accessible but we should always start by trying to get a working snippet of code first before we write our function.

Today's lab

```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)
```

If we want to get the average, we can use the `mean()` function

```r
mean(student1)
```

```
[1] 98.75
```

Let's be nice instructors by dropping the lowest score, so the answer here should be 100 Could use the `min()` function to find the lowest score:

```r
min(student1)
```

```
[1] 90
```

This isn't that helpful though, but `which.min()` is able to tell us the location of the lowest score:

```r
student1
```

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

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

```
[1] 8
```

```r
student1[8]
```

```
[1] 90
```

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

```
[1] 90
```

But we want to find the average of everything but the lowest score, use the minus sign!

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

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

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

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

Now the average of this gives the final grade with the dropped score:

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

```
[1] 100
```

Try it on student 2

```r
student2
```

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

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

```
[1] NA
```

Where's the problem?

```r
mean(student2)
```

```
[1] NA
```

Can't take the mean if NA is there, default mean is `na.rm=FALSE`, so change to `na.rm=TRUE`

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

```
[1] 91
```

Just dropped the NA How about student 3:

```r
student3
```

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

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

```
[1] 90
```

Dropped all the NA's! This score isn't fair

Want to stop working with `student1`, etc and typing it out each time, so let's work with the variable `x`

```r
x <- student2
```

We want to override the NA values, so if you miss a homework you score 0 on the homework Google and Claude told me about `is.na()`

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

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

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

```
[1] NA
```

We can use logicals to index a vector, example:

```r
y <- 1:5
y
```

```
[1] 1 2 3 4 5
```

```r
y>3
```

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

```r
y[y>3]
```

```
[1] 4 5
```

```r
y[y>3] <- 100
```

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

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

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

```
[1] 91
```

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

```
[1] 12.85714
```

Completed code:

```r
x <- student1
#mask NA values to 0
x[is.na(x)] <- 0
#drop lowest score and get the mean
mean(x[-which.min(x)])
```

```
[1] 100
```

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
grade <- function(x) {
#mask NA values to 0
x[is.na(x)] <- 0
#drop lowest score and get the mean
mean(x[-which.min(x)])
}
```

Use the function:

```r
grade(student3)
```

```
[1] 12.85714
```

We need to read the gradebook

```r
gradebook <- read.csv("https://tinyurl.com/gradeinput", row.names=1)
#make sure to use straight quotes "", rather than curly ones when reading a file
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
```

Use `apply()`

```r
#`array(x(array), margin(1=rows, 2=columns), fun(function))
student_grades <- apply(gradebook, 1, grade)
student_grades
```

```
 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
max(student_grades)
```

```
[1] 94.5
```

```r
which.max(student_grades)
```

```
student-18
        18
```

Top scoring student is student 18 with a score of 94.5

Q3.

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

```r
mask <- gradebook
mask[is.na(mask)] <- 0
mean_hw <- apply(mask, 2, mean)
mean_hw
```

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

```r
which.min(mean_hw)
```

```
hw2 
  2 
```

Homework 2 obtained the lowest scores overall.

Another way of doing it…

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

```
hw3
  3
```

This version eliminates all NA scores, so only those who did the homework are taken into account

We could do the sum

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

```
hw2
  2
```

Another different answer by using sum

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)?

Use the function `cor()`

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

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

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

```
hw5
  5
```

Homework 5 is the most correlated with overall student scores.