Week5_2(Making_a_function)

Roberto Ramos (PID: A59010570)

2/4/2022

Making R Functions

In this class we are going to learn all about functions in R.

First we will write a function to grade some student scores.

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)
```

```
mean(student1)
```

```
## [1] 98.75
```

```
#Add the "na.rm" argument to remove an NA values

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

```
## [1] 91
```

To find NA values, we found the `is.na()` fucntion.

```
student2
```

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

```
is.na(student2)
```

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

> Side-note: Logical vectors are used in R like:

```r
x <- 1:5

x < 5
```

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

`[]` asks R to return the values within a variable/dataset that meets the conditions within the brackets

```r
student2[is.na(student2)]
```

```
## [1] NA
```

Time to use a little place holder variable called "x"

```r
x <- student2

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

# This above says "set all NAs in x to 0"

x
```

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

```r
mean(x)
```

```
## [1] 79.625
```

```r
y <- student3

y[is.na(y)] <- 0

y
```

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

```r
mean(y)
```

```
## [1] 11.25
```

In this class, students are allowed to drop their lowest score for their final grade.

We can first use the `min()` function to find the lowest score. However, brining up the help page for `min()` with `?min`, we fine that we can use `min()` and `which.min()`

```r
#student 1

min(student1)
```

```
## [1] 90
```

```r
#returns the position in the vector that contains the min value

which.min(student1)
```

```
## [1] 8
```

```r
length(student1)
```

```
## [1] 8
```

Remember:`[]` asks R to return the values within a variable/dataset that meets the conditions within the brackets. So `x[-5]` would return everything in x except the 5 position

```r
x<- student1

# Return everything in "x" that isn't "-which.min(x)" (i.e. the 8th position in this case)
x[-which.min(x)]
```

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

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

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

```r
#copy student2 data to a new container
edit_student2 <- student2

#set NA values in new cotainer to 0
edit_student2[is.na(edit_student2)] <- 0

edit_student2
```

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

```r
#find the mean of student 2's grades after having omitted the lowest score with `-which.min()`
mean(edit_student2[-which.min(edit_student2)])
```

```
## [1] 91
```

We are ready to make this into a function called `grade()`

Every function in R has at least 3 things: - Name (grade) - Input arguments (student1) - Body (working code snipet)

```r
grade <- function(x){
  # set all NA values in x to 0
  x[is.na(x)] <- 0
  #Find the mean of all the values in x, having excluded the minimum value
  mean(x[-which.min(x)])
}
```

Now lets test it!

```
grade(student1)
```

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

```
grade(student2)
```

```
## [1] 91
```

```
grade(student3)
```

```
## [1] 12.85714
```

## Grade the class

To read a .csv file, use `read.csv()`:

```
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
```

We are going to learn about `apply()`. It's super useful!

```
apply(gradebook, 1, grade)
```

```
##  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 
```

> Q3: Using your grade() function and the supplied gradebook, Who is the top scoring student overall in the gradebook?

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

which.max(scores)
```

```
## student-18
##         18
```

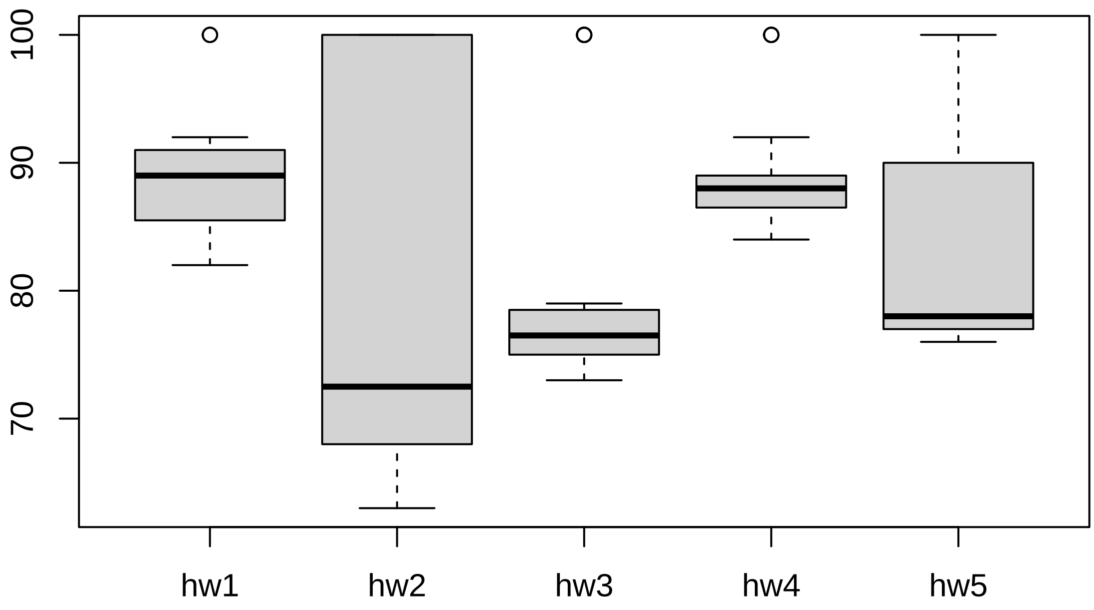
Student 18 is the top scoreing student

> Q5: From your analysis of the gradebook, which homework was toughest on students (i.e. obtained the lowest scores overall?

```
grades_NAremoved<- gradebook

grades_NAremoved[is.na(grades_NAremoved)] <- 0

hw_scores<- apply(gradebook, 2, grade)

which.min(hw_scores)
```

```
## hw2
##   2
```

```
boxplot(gradebook)
```



Homework 2 was the toughest on students

Optional Extension: From your analysis of the gradebook, which homework was most predictive of overall score (i.e. highest correlation with average grade score)?

```
cor(gradebook$hw1, scores)
```

```
## [1] 0.4250204
```

```
apply(grades_NAremoved, 2, cor, scores)
```

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

Homework 5 has the most predictive overall score.