

lab6 R functions

Diana Furlan

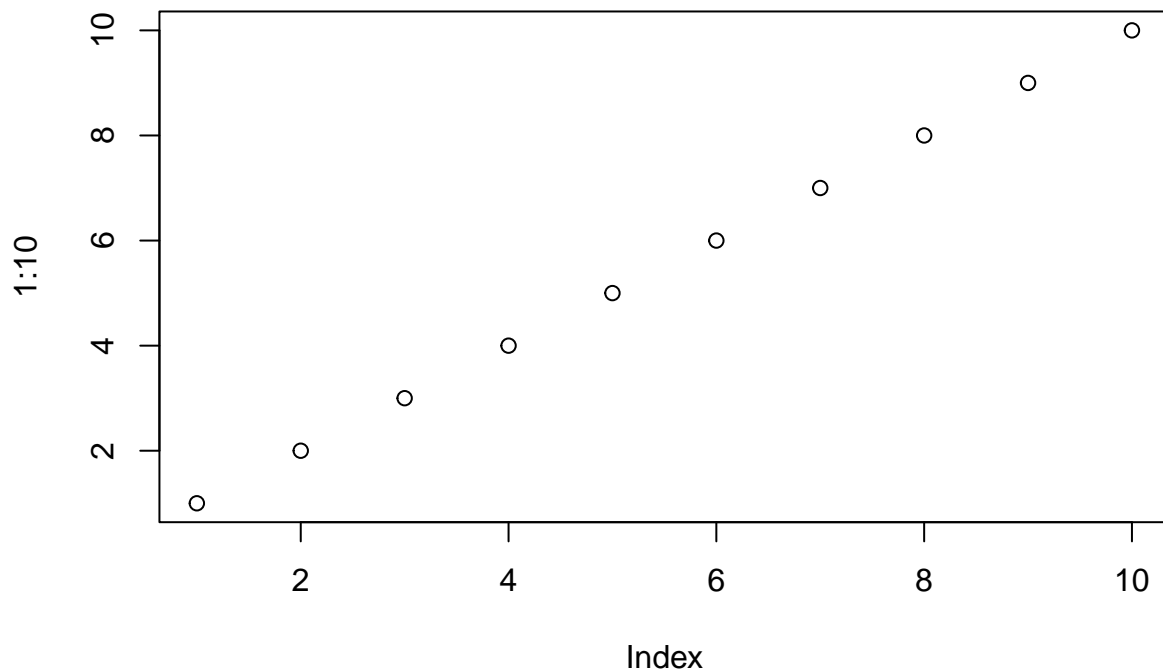
2024-10-21

This week we are introducing **R functions** and how to write our own functions

Questions to answer:

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 adequately 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]

```
plot(1:10)
```



```
# 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)
```

Follow the guidelines from class

-Write a working snippet of code that solves a simple problem

```
#Straight forward mean()
student1 <- c(100, 100, 100, 100, 100, 100, 100, 90)

mean(student1)
```

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

Drop the lowest score. First we need identify the lowest score.

```
#Which element of the vector is the lowest?
which.min(student1)
```

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

What I want is to now drop (i.e exclude) this lowest score from my mean() calculation.

```
#This will return everything but the eight value of the vector
student1[-8]
```

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

Now we can use the answer from which.min() to return all other elements of the vector

```
#This is our first working snippet
mean(student1[-which.min(student1)])
```

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

What about the other students?

We could try using na.rm=TRUE argument for mean but this is chirmol! unfair to students

```
student2 <- c(100, NA, 90, 90, 90, 90, 97, 80)
mean(student2, na.rm=TRUE)
```

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

```
student3 <- c(90, NA, NA, NA, NA, NA, NA, NA)
mean(student3, na.rm=TRUE)
```

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

Another approach is to replace all NA values with zero

First we need to find the NA elements of the vector

```
student2 <- c(100, NA, 90, 90, 90, 90, 97, 80)
x<- student2

is.na(x)
```

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

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

```
## [1] 2
```

Now we have id the NA elements we want to replace them w/zero.

```
#This wont replace the NA with Zero  
mean(x[-which(is.na(x))])
```

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

```
#this is useful
```

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

```
x
```

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

```
mean(x)
```

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

Drop the lowest score.

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

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

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

```
student3 <- c(90, NA, NA, NA, NA, NA, NA, NA)
```

```
x<-student3
```

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

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

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

Now we make our function

turn snippet into function needs to have 3 parts:

- A name, 'grade()'
- Input arguments, a vectore of student scores
- The body, (snippet code)

Using R studio select 'Code > Extract Function'

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

```
grade(student1)
```

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

```
grade(student2)
```

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

```
grade(student3)
```

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

This looks great, we now need to add comments :)

```
#' Calculate the average score for a vectore of  
#' student scores dropping the lowest score.  
#' Missing values will be treated as zero  
#'  
#' @param x A numeric vectore of hw scores
```

```

#'
#' @return Average score
#' @export
#'
#' @examples
#'
#' student <- c(100,NA,90,97)
#'
grade <- function(x) {
  #mask NA w/zero
  #treat missing values as zero

  x[is.na(x)]<- 0
  #Exclude lowest score from mean
  mean(x[-which.min(x)])
}

```

Using the function with whole class data from this: CSV format: “<https://tinyurl.com/gradeinput>”

```

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

```

```

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

```

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

We run the apply() function and save the results

```

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

```

```

## student-15 student-10 student-2 student-19 student-20 student-3 student-4
##      78.75      79.00      82.50      82.75      82.75      84.25      84.25
## student-11 student-9 student-14 student-17 student-5 student-6 student-16
##      86.00      87.75      87.75      88.00      88.25      89.00      89.50
## student-1 student-12 student-13 student-8 student-7 student-18
##      91.75      91.75      92.25      93.75      94.00      94.50

```

```

which.max(results)

```

```

## student-18
##      18

```

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

```

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
```

```
ave.scores <- apply(gradebook, 2, mean, na.rm=TRUE)
ave.scores
```

```
##      hw1      hw2      hw3      hw4      hw5
## 89.00000 80.88889 80.80000 89.63158 83.42105
```

```
which.min(ave.scores)
```

```
## hw3
## 3
```

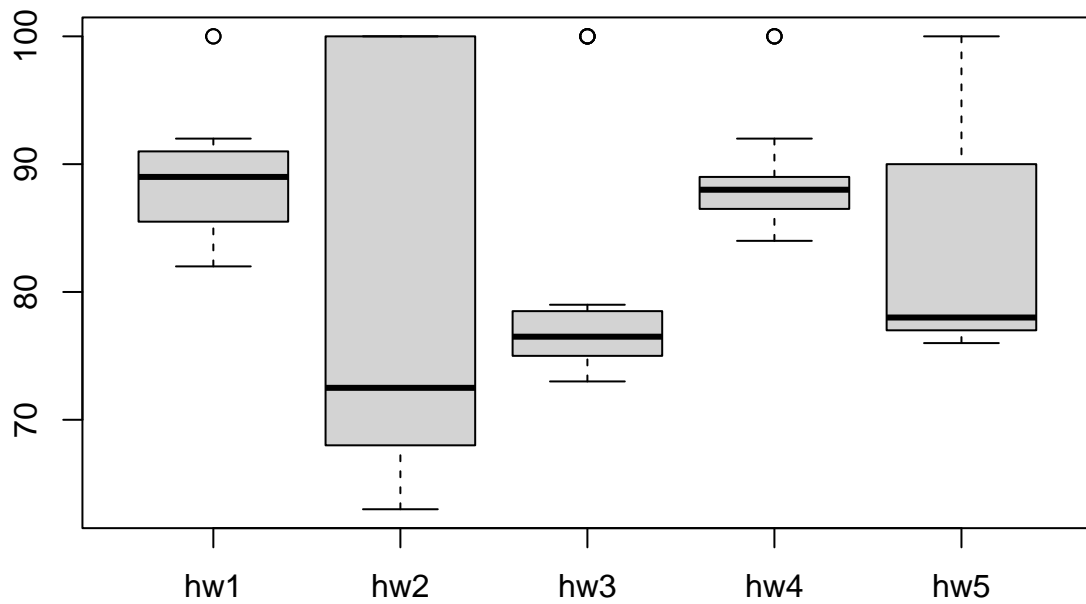
```
med.scores <- apply(gradebook, 2, median, na.rm=TRUE)
med.scores
```

```
## hw1 hw2 hw3 hw4 hw5
## 89.0 72.5 76.5 88.0 78.0
```

```
which.min(med.scores)
```

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

```
boxplot(gradebook)
```



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]

Q5. Make sure you save your Quarto document and can click the “Render” (or Rmarkdown”Knit”) button to generate a PDF foramt report without errors. Finally, submit your PDF to gradescope. [1pt]

Knit the document to make a PDF