Lab 6

Xiaoyu Lian (A17025943)

Today we are going to explore R functions and begin to think about writting 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.

```
add <- function (x,y=1, z=0) {x+y}
```

Try

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

```
[1] 11
```

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

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

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

```
[1] 20
```

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

```
[1] 20
```

```
#add('Lorraine')
```

#Lab 6 >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" [3pts]

```
# 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 calculating the average for student

```
mean(student1)
```

```
[1] 98.75
```

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

```
[1] 91
```

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

```
[1] 90
```

need to deal with the NA score later drop the lowest score from a given student 1. use `min()` and `which.min()` function

```
min(student1)
```

```
[1] 90
```

```
#find the location of the min value not the value itself, use which.min()
which.min(student1)
```

```
[1] 8
```

```
#put together
student1[-which.min(student1)]
```

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

what if there're more than one equally low score

```
a <- c(100,90,100,90,100,100)
which.min(a)
```

```
[1] 2
```

```
a[-which.min(a)]
```

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

```
removelow <- function(studentid){studentid[-which.min(studentid)]}
removelow(student1)
```

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

how to deal with NA value? one is make NA equal to zero

```
student2
```

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

```
#is.na set NA value to true, ! is to flip the logical, let true-->false
is.na(student2)
```

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

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

```
[1] NA
```

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

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

```
#find the NA value and make it 0
student2[is.na(student2)]<-0
```

```
#generalize the function()
grade <- function(studentid){
  #find NAs and make them 0
  studentid[is.na(studentid)] <- 0
  #drop lowest value and find mean
  mean(studentid[-which.min(studentid)])
}

grade(student1)
```

```
[1] 100
```

```
grade(student2)
```

```
[1] 91
```

```
grade(student3)
```

```
[1] 12.85714
```

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

```
           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 `grade_book` data set, I need to decide whether I want to apply to the `grade()` function over the rows (1) or columns (2) of the `grade_book`

```
#apply(X, margin, function), x= data, margin: 1= row, 2 = column
mean_student <- apply(grade_book, 1, grade)
mean_student
```

```
 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?

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

```
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]

```
#this will drop the lowest score
#mean_hw <- apply(grade_book,2,grade)
#mean_hw[which.min(mean_hw)]
apply(grade_book, 2, mean, na.rm = T)
```

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

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

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

I can modify `grade()` function to do this too -ie not drop the lowest options

```
#generalize the function()
grade <- function(studentid, drop.low = TRUE){
  studentid[is.na(studentid)] <- 0
  if(drop.low){output <- mean(studentid[-which.min(studentid)])}
  else{output <- mean(studentid)}
  return(output)
}
grade(student2, TRUE)
```

```
[1] 91
```

```
grade(student2, FALSE)
```

```
[1] 79.625
```

> 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 correlation in R is called `cor()`

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

```
[1] 1
```

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

```
[1] -0.6822423
```

```r
cor(mean_student, masked_gradebook)
```

```
           hw1       hw2       hw3       hw4       hw5
[1,] 0.4250204 0.176778 0.3042561 0.3810884 0.6325982
```

if want to use `apply()` function over the `masked_gradebook` and use the `mean_student` scores for the class

```r
apply(masked_gradebook, 2, cor, y = mean_student)
```

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