Lab 6: R Functions

Gretel Warmuth (PID A17595945)

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

Let's write our first function to add numbers.

Every function in R has at least 3 things: ?plot - a **name** (we pick this) - one or more inputs- **arguments** - the **body** (where the work happens)

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

Let's try it

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

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

```
add(10)
```

```
[1] 11
```

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

```
[1] 20
```

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

```
[1] 30
```

```
# mean( c(10, 10, NA) na.rm = T)
```

Lab Sheet Work

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"

```
# 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)
```

Calculating the average of student1:

```
student1
```

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

```
mean(student1)
```

```
[1] 98.75
```

Calculating the average on student2:

```
student2
```

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

```
mean(student2)
```

```
[1] NA
```

Creating a mean for student2:

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

```
[1] 91
```

Calculating the average for student3:

```
student3
```

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

```
mean(student3)
```

```
[1] NA
```

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

```
[1] 90
```

We need to calculate an average for all students where their lowest score is neglected and "NA"s count as 0s.

```
student1
```

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

```
student1[-8]
```

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

The `min()` function can help us find the lowest score

```
min(student1)
```

```
[1] 90
```

I need to find the location of the minimum value, not the value itself. I can use `which.min()`

```
student1
```

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

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

```
[1] 8
```

Put it together to create a new average

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

```
[1] 8
```

```
student1[-8]
```

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

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

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

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

```
[1] 100
```

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

```
[1] 100
```

We could make all NA values equal to 0

```
x <- student2
x
```

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

```
x[2] <- 0
x
```

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

```
x <- student2
x
```

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

```
x[is.na(x)] = 0
x
```

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

We've found a woking snippet:

```
## Find NAs in `x` and make them 0
x <- student3
x[is.na(x)] <- 0

## Finds the mininmum value and rm's it before getting mean
mean(student1[-which.min(student1)])
```

```
[1] 100
```

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

```
[1] 12.85714
```

Lets make it into a function:

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

For each student:

```
grade(student1)
```

```
[1] 100
```

```
grade(student2)
```

```
[1] 91
```

```
grade(student3)
```

```
[1] 12.85714
```

Now `apply()` to our class gradebook

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

To use the `apply()` function on this `gradebook` dataset I need to decide whether I want to "apply" the `grade()` function over the rows or columns of the `gradebook`

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

```
 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?

```
which.max(ans)
```

```
student-18
        18
```

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

```
student-18
      94.5
```

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

```
apply(gradebook, 2, grade)
```

```
     hw1      hw2      hw3      hw4      hw5
89.36842 76.63158 81.21053 89.63158 83.42105
```

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

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

```
masked_gradebook <- gradebook
masked_gradebook[is.na(masked_gradebook)] = 0
masked_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   0  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   0  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   0
student-16  92 100  74  89  77
student-17  88  63 100  86  78
student-18  91   0 100  87 100
student-19  91  68  75  86  79
student-20  91  68  76  88  76
```

```
apply(masked_gradebook, 2, mean)
```

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

I could modifty the `grade()` function to do this too- i.e not drop the lowest option

```
grade2 <- function(x, drop.low= TRUE) {
  x[is.na(x)] <- 0

  if(drop.low) {
    cat("Hello Low")
    out <-  mean(x[-which.min(x)])
  } else {
  out <- mean(x)
  cat("No Low")
  }
  return(out)
}
grade2(student1, TRUE)
```

```
Hello Low
```

```
[1] 100
```

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 look at correlations is called `cor()`

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

cor(x,y)
```

```
[1] 1
```

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

```
[1] -0.6822423
```

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

```
[1] 0.4250204
```

```
cor(ans, masked_gradebook$hw1)
```

```
[1] 0.4250204
```

```
cor(ans, masked_gradebook$hw2)
```

```
[1] 0.176778
```

```
cor(ans, masked_gradebook$hw3)
```

```
[1] 0.3042561
```

```
cor(ans, masked_gradebook$hw4)
```

```
[1] 0.3810884
```

```
cor(ans, masked_gradebook$hw5)
```

```
[1] 0.6325982
```

I want to `apply()` the `cor()` function ove the `masked_gradebook` and use the `ans` scores for the class

```
apply(masked_gradebook, 2, cor, y = ans)
```

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

```
apply(masked_gradebook, 2, cor, ans)[which.max(apply(masked_gradebook, 2, cor, ans))]
```

```
      hw5
0.6325982
```