Writing_Function_Class_06

Johann Tailor

2024-01-06

Our first simple functions:

All functions have three parts:

- A name
- Input arguments (none, one or more)
- A body

A function to add a number:

Here: (x, y) is the data input and {x + y} is the function

```
sillyadd <- function(x, y) { x + y }
```

Let me try out this function

```
sillyadd(100, 100)
```

```
[1] 200
```

I added both codes into one:

```
sillyadd <- function(x, y=1) { x + y }
sillyadd(100)
```

```
[1] 101
```

Let's do some other activities:

Solve the following:

Question 1:

> 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]

Find average for each student **after dropping the lowest score**:

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)

The following is my work:

My codes:

```r
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)

sorted_grade <- sort(student1)

all_students <- data.frame(student1, student2, student3)

all_students
```

```
  student1 student2 student3
1      100      100       90
2      100       NA       NA
3      100       90       NA
4      100       90       NA
5      100       90       NA
6      100       90       NA
7      100       97       NA
```

```
8        90        80        NA
```

This is for student 1:

```r
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)

which.min(student1)
```

```
[1] 8
```

```r
#tells where is the lowest score:
student1[-8]
```

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

```r
lowest_dropped <- student1[-8]
#mean after dropping the lowest one:
mean(lowest_dropped)
```

```
[1] 100
```

Let's see if it works for student 2

I had to add the na.rm to remove:

```r
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)

which.min(student2)
```

```
[1] 8
```

```r
#tells where is the lowest score:
student2[-8]
```

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

```r
lowest_dropped <- student2[-8]
#mean after dropping the lowest one:
mean(lowest_dropped, na.rm = TRUE)
```

```
[1] 92.83333
```

Let's see if it works for student 3

I had to add the na.rm to remove NA values:

```r
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)

which.min(student3)
```

```
[1] 1
```

```r
#tells where is the lowest score:
student3[-8]
```

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

```r
lowest_dropped <- student3[-8]
#mean after dropping the lowest one:
mean(lowest_dropped, na.rm = TRUE)
```

```
[1] 90
```

Find and replace the Na values with zero:

```r
#simple function:
 x <- student3
x
```

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

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

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

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

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

```
x <- student2

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

```
[1] 91
```

Converting it into a function:

Answer to QUESTION 1:

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


grade(student3)
```

```
[1] 12.85714
```

Now, lets use this to analyze a data set:

Read the following file: https://tinyurl.com/gradeinput

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

Now, lets use the function we generated to analyze the gradebook

```
#we can "apply" our new 'grade()' function over the rows and coloumns with Margins
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 
```

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

Question 2:

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

Answer to QUESTION 2:

```
which.max(results)
```

```
student-18 
        18 
```

Answer 2: Student-18

Question 3:

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

Answer to QUESTION 3:

For the apply function, here are some parameters:

Margin 1 = Rows Margin 2 = Col

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

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

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

```
hw3 
  3 
```

Answer 3: Homework 3

Let's see how well we are teaching the class by looking at the correlation of overall grades and assignments:

Question 4:

> From your analysis of the gradebook, which homework was most predictive of overall score (i.e. highest correlation with average grade score)? [1pt]

Answer to QUESTION 4:

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

```
          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
```

```r
cor(mask$hw5, results)
```

```
[1] 0.6325982
```

```r
#apply the function to whole dataframe:

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

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

Answer 4: Homework 5