Class06: R Functions

Rahul Nedunuri (PID:A16297840)

2024-01-25

R Functions

Functions are how we get things done. We can call functions to avoid redundant code. Writing functions in R is doable.

All functions in R have at least three things: - **name** which we choose - **input args** aka the input to our function - **body** the main chunk of code

`#| eval: false` will print code when rendered but won't run it if there is an error

Silly first function to add 2 nums:

```
addme <- function(in1, in2=1) {
  in1 + in2
}
```

Lab for today

```
# 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
```

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

```
[1] 91
```

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

```
[1] 90
```

This is not useful, student 3 shouldn't have an overall score of 90 if they missed 7 assignments. Problem with how we are dealing with NA

```
ind <- which.min(student1)
ind
```

```
[1] 8
```

8th index is position of minimum in student1's grades.

Calculating mean after removing 8th index with `[-which.min()]`

```
mean(student1[-ind])
```

```
[1] 100
```

We found student1's grade after the drop.

Now we can use a common shortcut and use `x` as the input.

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

```
[1] 100
```

We still have the problem of missing values. One idea is to replace NA values with 0.

```
y <- 1:5
y[y==3] <- NA
is.na(y)
```

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

How can I remove NA? I can flip logicals with `!`

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

Let's try it on student grades.

```
  x <- student3

  #Change NA values to 0
  x[is.na(x)] <- 0
  mean( x[ -which.min(x)])
```

```
[1] 12.85714
```

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"

```
  # function grade(): determines grade by dropping the lowest single score
  grade <- function(x) {
    #Replace all NA values with zeroes
    x[is.na(x)] <- 0
    #Find mean after dropping min score
    mean(x[-which.min(x)])
  }
```

```
  # Read data from link to csv format
  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
```

Apply the `grade()` function to all students

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

Q2

Using your grade() function and the supplied gradebook, Who is the top scoring student overall in the gradebook?
Ans: STUDENT18

```
highest = which.max(final)
print(highest)
```

```
student-18
        18
```

```
max(final)
```

```
[1] 94.5
```

Q3

From your analysis of the gradebook, which homework was toughest on students (i.e. obtained the lowest scores overall?
Ans: HW2 was the hardest using the sum of scores and dropping NA

```
hw <- apply(gradebook, 2, sum, na.rm=T)
which.min(hw)
```

```
hw2
  2
```

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)?
Ans: HW5

```
# Make all NA values --> 0
mask <- gradebook
mask[ is.na(mask) ] <- 0
```

We can use the `cor()` function for correlation analysis and use the apply function.

```
apply(mask, 2, cor, final)
```

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