Class 6: R Functions

Stefanie Hodapp (PID: A53300084)

10/15/2021

Quick Rmarkdown intro

We can write text of course just like any file. We can **style text to be bold** or *italic*.

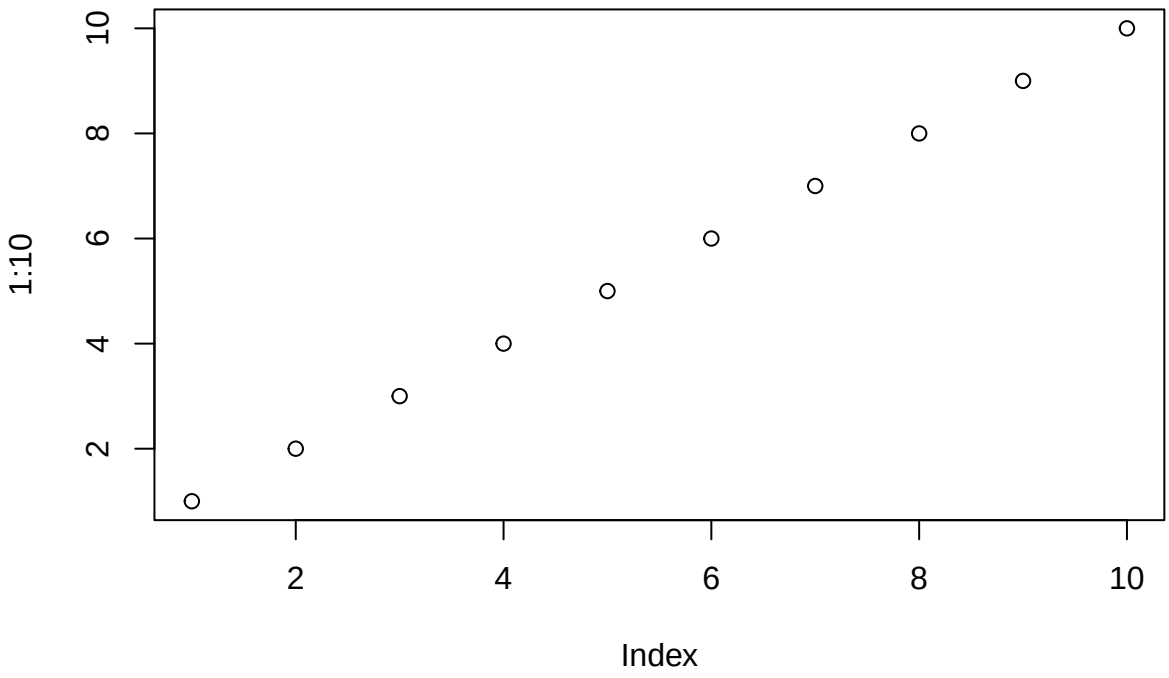
Do :

- this
- and that
- and another thing

This is more text
and this is a new line

We can include some code:

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



Time to write a function

```
# 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)
```

First I want to find the lowest score. I can use the **min()** to find it and the **which.min()** function to find where it is (i.e. its position in the vector)

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

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

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

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

Now I can call the **mean()** function to get the average.

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

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

Does this work for student2?

```
mean(student2[ -which.min(student2) ], na.rm=TRUE)
```

```
## [1] 92.83333
```

One great idea is to replace the na values with zero. Let's do it.

```
which(is.na(student2))
```

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

This is.na() function returns a logical vector where TRUE elements indicate thepresence of NA values.

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

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

Let's replace NAs with 0 and print the new values in this

```
student2_new <- student2
student2_new[is.na(student2_new)] = 0
student2_new
```

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

```
mean(student2_new[ -which.min(student2_new)])
```

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

How about student 3?

```
student3_new <- student3
student3_new[is.na(student3_new)] = 0
student3_new
```

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

```
mean(student3_new[ -which.min(student3_new)])
```

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

We can make the object names more clear

```
x <- student3
x[is.na(x)] = 0
mean(x[ -which.min(x)])
```

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

A vector containing an accidental character can be changed to a numeric vector using the following function:

```
student4 <- c(100, NA, 90, "90", 90, 90, 97, 80)
```

```
x <- student4
x <- as.numeric(x)
x[is.na(x)] = 0
mean(x[ -which.min(x)])
```

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

Now we can write our function: All functions have at least 3 things. A name, an input arguments, and a body.

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

```
grade(student1)
```

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

Develop a function for calculating average grades for fictional students in a fictional class.

Inpur CSV file containing the class gradebook

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

```
##            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
```

Write a function grade() to determine an overall grade from a vector of student homework assignment scores dropping the lowest single score.

We are going to use the super useful **apply()** function to grade all the students with our **grade()** function.

Question 1

```
Q1 <- apply(scores, 1, grade)
Q1
```

```
##  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
```

Question 2

```
Q2 <- which.max(Q1)
Q2
```

```
## student-18
##         18
```

Q3. Replace or mask Na values to zero

```
mask <- scores
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
```

Question 3. Ignore the NA missing values with na.rm=TRUE

```
Q3 <- apply(mask, 2, mean, na.rm=TRUE)
Q3
```

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

Question 4. From your analysis of the gradebook, which homework was most predictive of overall score (i.e. highest correlation with average grade score)?

Here we will use the **cor()** function.

```
cor(mask$hw1, Q1)
```

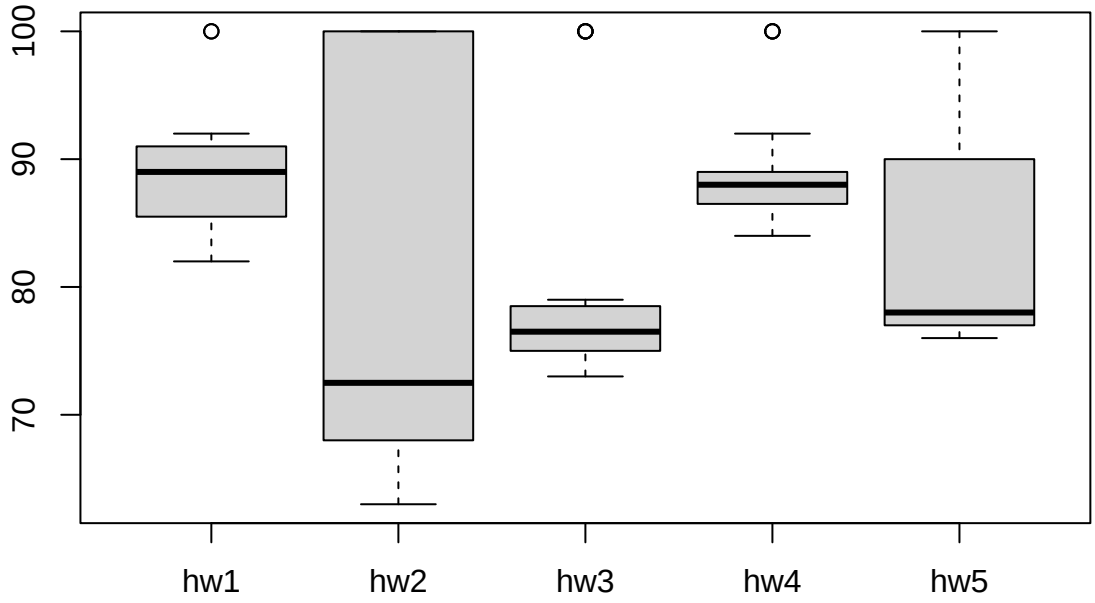
```
## [1] 0.4250204
```

I can call the **cor()** for every homework and get a value for each but that sucks. Let's use **aply()** and do them all in one go.

```
apply(mask,2,cor, Q1)
```

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

```
boxplot(scores)
```


100
90
80
70
hw1
hw2
hw3
hw4
hw5