Class 6: R Functions

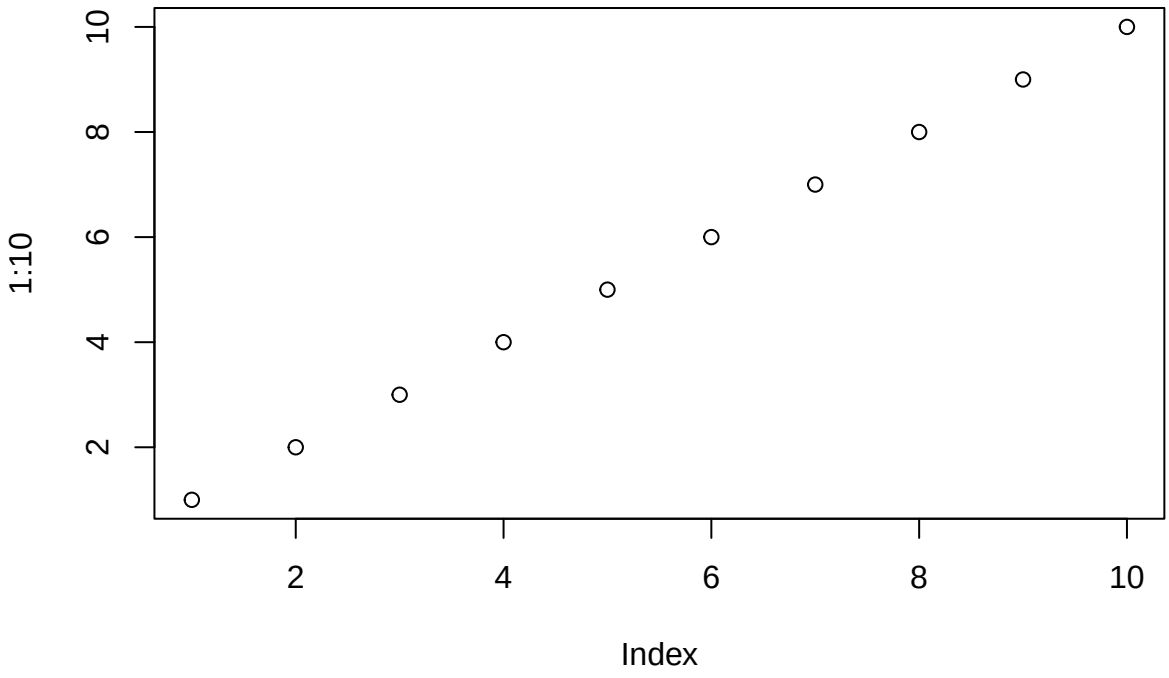
Hope (PID: A15652616)

10/14/2021

A play with Rmarkdown

This is some plain text. I can make things **bold**. I can also make things *italic*.

```
# This is a code chunk
plot(1:10)
```



R functions

In today's class we are going to write a function together that grades some student work.

Question for today:

Q1. Write a function grade() to determine an overall grade from a vector of student homeworkassignment 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)
```

Let's start with student1 and find their average score.

```
mean(student1)
```

```
## [1] 98.75
```

But we want to drop the lowest score... We could try the **min()** function

```
min(student1)
```

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

We can also try **which.min()**

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

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

This gives the postition of the lowest score

```
# This would be the lowest score
student1[which.min(student1)]
```

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

To drop this value I can use minus

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

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

Let's use mean() again to get the average minus the lowest score.

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

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

Start with student 2

```
mean(student2[-which.min(student2)])
```

```
## [1] NA
```

This doesn't work. It gives NA if a student is missing a homework. We need to remove the NA elements of the vector.

Lets try using something else

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

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

This is not what we want. It dropped the 80 instead of the NA which is a missing homework assignment.

Lets try student3 really quick:

```
mean(student3[ -which.min(student3) ], na.rm=TRUE)
```

```
## [1] NaN
```

It removed the lowest number which was 90 and it is trying to take the mean of all of the NA's and that doesn't work.

What if we make all of the NA's in the grades into a zero. Let's try with student2. To do this we can try **is.na()**

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

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

The **is.na()** function returns a logical vector where TRUE elements represent where the NA values are.

To use **is.na()** we can ask which position it is.

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

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

To make the NA value into a zero:

```
student.prime <- student2
student.prime
```

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

```
student.prime[which(is.na(student2)) ] = 0
student.prime
```

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

Now we need to put this all together to get the average score dropping the lowest score where we map NA values to zero.

```
student.prime <- student2
student.prime
```

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

```
student.prime[which(is.na(student2)) ] = 0
student.prime
```

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

```
mean(student.prime[-which.min(student.prime)])
```

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

Lets make sure this is correct by taking out the zero on our own.

```
student.prime
```

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

```
mean(c(100, 90, 90, 90, 90, 97, 80))
```

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

We got the same thing as before!

Lets try this for student 3

```
student.prime <- student3
student.prime
```

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

```
student.prime[which(is.na(student3)) ] = 0
student.prime
```

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

```
mean(student.prime[-which.min(student.prime)])
```

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

We have a working snippet. Let's simplify by changing all of the student.prime's to x

```
x <- student3
# Map NA values to 0
x[which(is.na(x)) ] = 0
# Find the mean without the lowest score
mean(x[-which.min(x)])
```

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

Now we can use this as the body of my function.

```
grade <- function(x) {
# Make sure our scores are all numbers
  x <- as.numeric(x)
# Map NA values to 0
x[which(is.na(x)) ] = 0
# Find the mean without the lowest score
mean(x[-which.min(x)])
}
```

The function is made, lets try to get a grade for student 1.

```
grade(student1)
```

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

Lets try student 2.

```
grade(student2)
```

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

Last we can try for student3

```
grade(student3)
```

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

Now we can read the full grade book CSV file.

```
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
```

```
grade(scores[10,])
```

```
## [1] 79
```

Now that this works we need to apply to all scores by using the **apply()** function.

```
apply(scores,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
```

To find the lowest score:

```
which.min(apply(scores,1,grade))
```

```
## student-15
##         15
```

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

To find the max score:

```
which.max(apply(scores,1,grade))
```

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

> Q3. From your analysis of the gradebook, which homework was toughest on students (i.e. obtained the lowest scores overall? [2pts]

To find which homework was the hardest for the students:

```
apply(scores,2, mean, na.rm=TRUE)
```

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

So hw 3 was the hardest for the students.