

Class06: R Fucntions

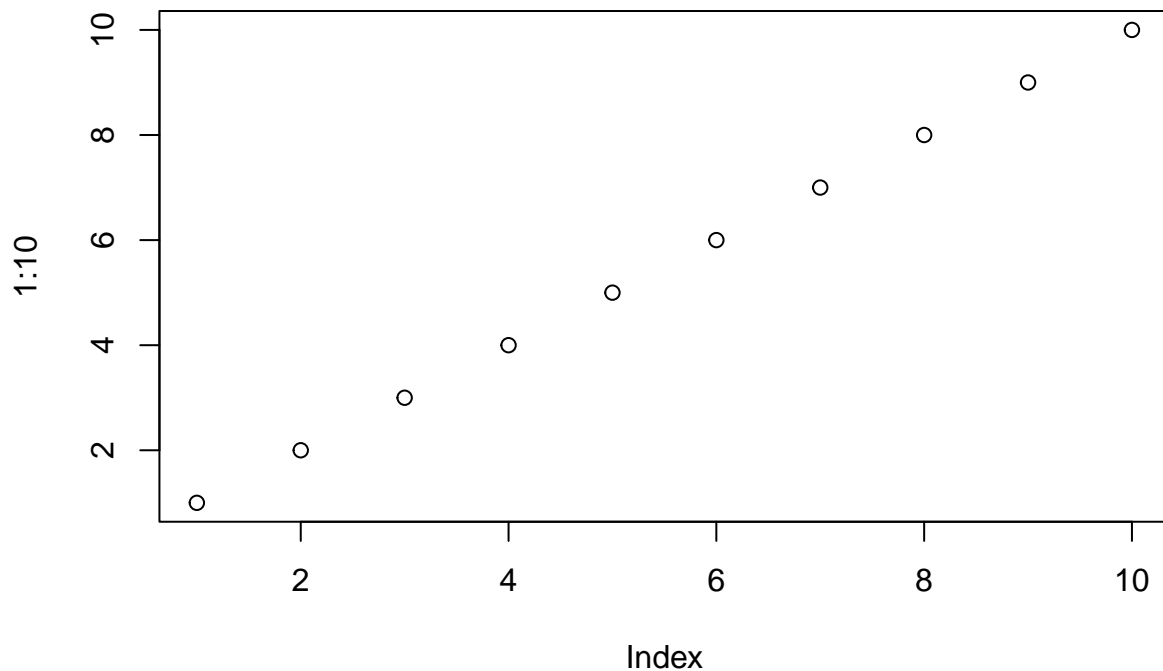
Tom Quach (A15549142)

10/14/2021

A play with Rmarkdown

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

Questions for today: > **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 adequately

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 **mins()** function

```
min(student1)
```

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

The **which.min()** function looks useful:

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

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

To drop the lowest score for student one, we can use minus

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

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

Let’s now use **mean()** to get the average minus the lower score

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

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

```
student2
```

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

Let’s use the same code from student1 to find the average minus the lowest score for student2

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

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

Well this sucks! It gives NA if a student has a missing homework

We need to remove the NA elements of the vector

```
which.min(student2)
```

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

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

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

The average above for student2 is a result from dropping the 80 (the lowest number and not the NA, the missing homework)

Let's take a look at student3

```
student3
```

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

Average minus the lowest score for student3

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

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

We can replace the NA (missing homework) with zero Let's try this with student2

```
student2
```

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

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

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

The **is.na()** function returns a logical vector where TRUE elements represent where the NA values are
Finding where NA is located on the student2 vector

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

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

Now let's make the NA values into zeros

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

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

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

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

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

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

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

Let's try this with student3

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

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

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

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

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

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

We got our working snippet! Lets simplify

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

```
grade(student2)
```

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

Now read the full gradebook CSV file

```
scores <- read.csv("https://tinyurl.com/gradeinput", row.name=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[2,])
```

```
## [1] 82.5
```

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

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

```
is.numeric(student1)
```

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

```
is.numeric(scores[10,])
```

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

```
as.numeric(c(1,2,NA,4,5))
```

```
## [1] 1 2 NA 4 5
```

Now grade all students 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
```

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

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

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

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

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

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

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

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

```
which.min(ans)
```

```
## hw3
##    3
```