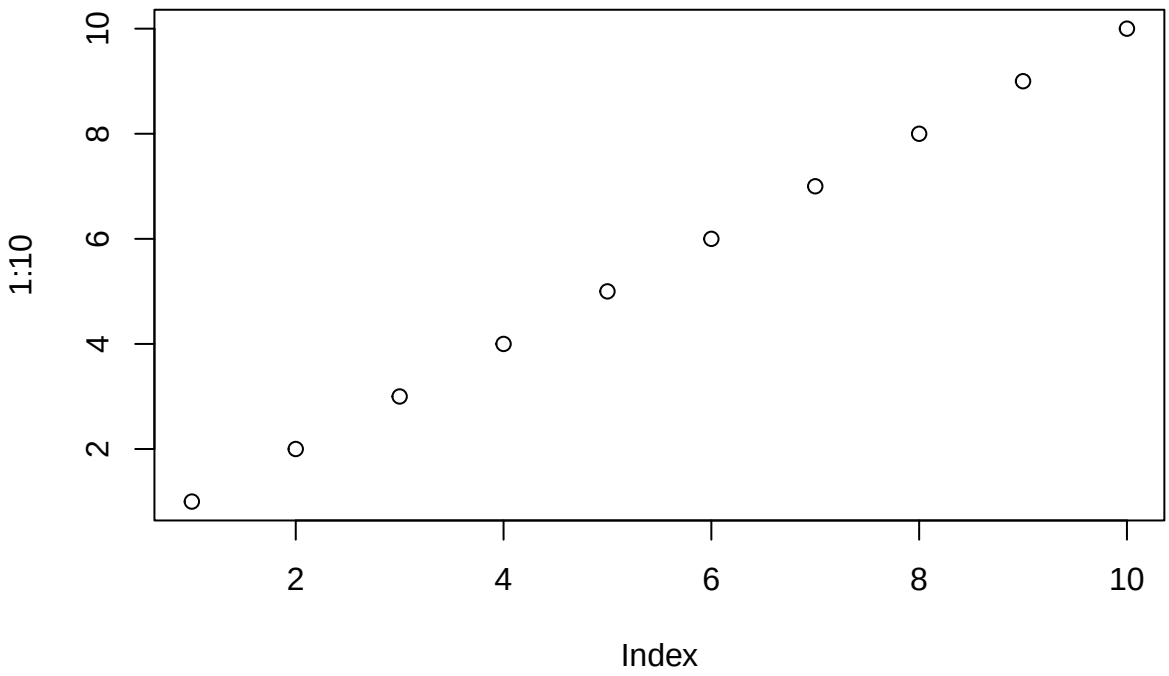
Class 6: R Functions

Camryn McCann (PID: A15437387)

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 students 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 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]

```r
#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.

```r
mean(student1)
```

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

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

```r
min(student1)
```

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

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

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

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

Cool! This gave us the position of the lowest score

Now we can remove this value using - (everything but)

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

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

Now we can put this into the mean() to get average score when dropping the lowest score!

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

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

This will not work for student 2 because the NA does not hold value.

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

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

We need to remove the NA elements from the vector.

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

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

Oh no! This is not what we want, the function dropped the 80 (i.e the lowest numerical value) and not NA (i,e the missing HW)

Now let's check student 3 as well

```
student3
```

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

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

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

Ugh! We run into a similar problem with student3...

One new idea is to give NA a numerical value: **0** Let's go back and look at student2

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

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

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

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

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

Now that we located NA values, we can convert them to 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
```

Let's put it all together to get average score, dropping the lowest where we map NA values to zero.

```
student.prime <- student2
student.prime[which(is.na(student.prime))]=0
mean(student.prime[-which.min(student.prime)])
```

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

It worked! Now, we need to see if it works for student 3, to determine if this would be helpful in a function for all the students in our gradebook.

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

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

Perfect! Now its time to rewrite for clarity!

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

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

Now we can use this as the body of our function

```
grade <- function(x){
  #make sure all the scores are read as numbers
  x <- as.numeric(x)
  # Map NA values to zero
  x[which(is.na(x))]=0
  #Then find the mean without the lowest value
  mean(x[-which.min(x)])}
```

Let's test it!

```
grade(student1)
```

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

```
grade(student2)
```

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

```
grade(student3)
```

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

Next, let's apply our function to the full gradebook 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
```

Test it on just one student first

```
grade(scores[3,])
```

```
## [1] 84.25
```

It originally was not working because R is reading the numbers as non-numeric, so we needed to edit our grade function to force read the file as numbers. (already went back to original and edited which is why it works)

Finally, lets apply them to the entire table using **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
```

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

```
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]

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

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

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

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

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)? [1pt]

```
mask <- scores
mask[is.na(scores)]=0
cor(apply(scores,1,grade), mask)
```

```
##            hw1      hw2       hw3       hw4       hw5
## [1,] 0.4250204 0.176778 0.3042561 0.3810884 0.6325982
```

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

```
## [1] 5
```