functionsintro

Nicole Jacobson A13081206

R markdown intro

Intro to functions

```
rescale <- function(x) {
  rng <- range(x)
(x-rng[1])/(rng[2]-rng[1])
}

rescale
```

```
## function(x) {
##   rng <- range(x)
## (x-rng[1])/(rng[2]-rng[1])
## }
```

```
rescale(1:10)
```

```
##  [1] 0.0000000 0.1111111 0.2222222 0.3333333 0.4444444 0.5555556 0.6666667
##  [8] 0.7777778 0.8888889 1.0000000
```

```
rescale(c(1,4,10,20))
```

```
## [1] 0.0000000 0.1578947 0.4736842 1.0000000
```

> 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]

```
# 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)
```

```
# tells us the position at which the minimum value is found
which.min(student1)
```

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

```
# - in the front will do everything but that is listed
# add mean at the front to get the mean of those values
mean(student1[ - which.min(student1)])
```

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

```
# Gives us the position where we find an NA
which(is.na(student2))
```

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

```
# Let's replace the NAs with zeros.using the [] we are making the positions with an NA 0
student.prime <- student3
student.prime[is.na(student.prime)] = 0
student.prime
```

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

```
#put pieces together to get the final function
mean(student.prime[ - which.min(student.prime)])
```

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

```
# now we can make our function. We will take out working piece and make it a function (code -> extract
grade <- function(student.prime) {
  student.prime[is.na(student.prime)] = 0
  mean(student.prime[ - which.min(student.prime)])
}
```

```
grade(student1)
```

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

```
grade(student2)
```

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

```
grade(student3)
```

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

```
#' Calculate the average score for a class of scores excluding the lowest score. Missing values are tre
#'
#' @param student.prime numeric vector of homework scores
#'
#' @return average score
#' @export
#'
```

```r
#' @examples student <- c(100, NA, 90, 80)
#' grade(student)
grade <- function(student.prime) {
  # made the NA values equal to zero
  student.prime[is.na(student.prime)] = 0
  # made to exclude the lowest value
  mean(student.prime[ - which.min(student.prime)])
}
```

```r
# now we can take the grade book and grade the class (example class grade book found here https://tinyu

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

```r
# apply(x, margin, fun, ...) x is the dataset, margin 1 indicates rows 2 indicates columns or both c(1,
apply(gradebook, 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
```

```r
# who is the top scoring student?
results <- apply(gradebook, 1, grade)
sort (results, decreasing = TRUE)
```

```
## student-18  student-7  student-8 student-13  student-1 student-12 student-16
```

```
##      94.50      94.00      93.75      92.25      91.75      91.75      89.50
##  student-6  student-5 student-17  student-9 student-14 student-11  student-3
##      89.00      88.25      88.00      87.75      87.75      86.00      84.25
##  student-4 student-19 student-20  student-2 student-10 student-15
##      84.25      82.75      82.75      82.50      79.00      78.75
```

```
# another option
which.max(results)
```

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

```
# was was the toughest assignment? Which column has the lowest score? Two options.
hw.mean <- apply(gradebook, 2, mean, na.rm = TRUE)
which.min(hw.mean)
```

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

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

```
## hw2
##   2
```

```
boxplot(gradebook)
```