IST 687
Homework 4
Due Date: 11/2

Code requires the following packages to run

```
library(moments) #install.packages("moments")
library(scales) #install.packages("scales")
```

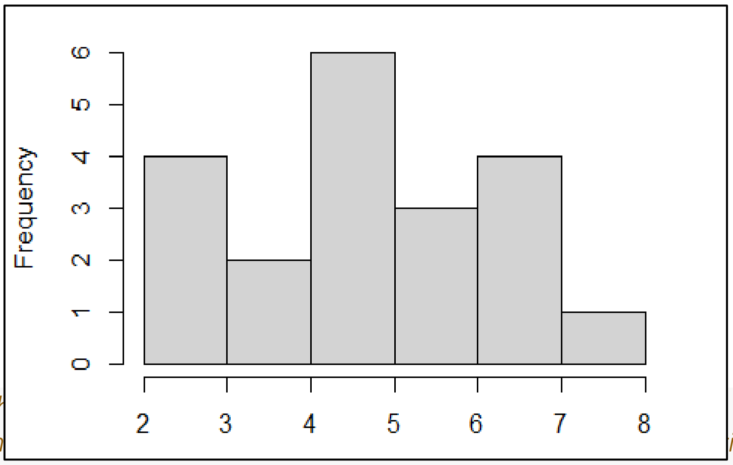
Step 1: Write a summarizing function to understand the distribution of a vector

```
#create function that calculates descriptive characteristics for a vector
printVecInfo <- function(vector){
print(mean(vector)) #a. mean
print(median(vector)) #b. median
print(min(vector)) #c. min
print(max(vector)) #d. max
print(sd(vector)) #e. st dev
print(quantile(vector, probs = c(0.05, 0.95))) #quantiles
print(skewness(vector))} #skewness

#test the function with a vector
testVector1 <- c(1,2,3,4,5,6,7,8,9,10,50)
printVecInfo(testVector1)
## [1] 9.545455
## [1] 6
## [1] 1
## [1] 50
## [1] 13.72125
##   5%  95%
##  1.5 30.0
## [1] 2.620396
```

Step 2: Creating Samples in a Jar

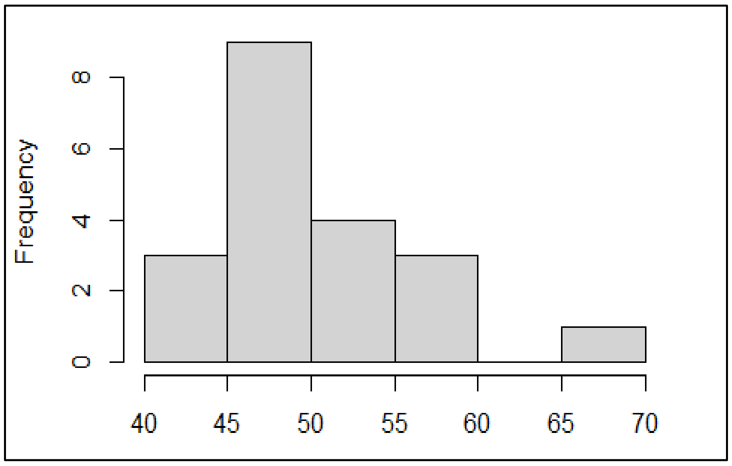
```
#4. create a variable 'jar' that has 50 red and 50 blue marbles
jarRed <- replicate(50, "red", simplify = TRUE)
jarBlue <- replicate(50, "blue", simplify = TRUE)
jar <- c(jarRed, jarBlue)
#5. confirm there are 50 reds by summing the samples that are red
sum(jar=="red")
## [1] 50
```

```r
#6. Sample 10 'marbles' (really strings) from the jar. How many are red? What was the percentage of red marbles?
sum(sample(jar, 10, replace=TRUE)=="red") #Sum total number of red marbles from our sample
## [1] 4
length(sample(jar, 10, replace=TRUE)) #total number of marbles in the sample
## [1] 10
percent(sum(sample(jar, 10, replace=TRUE)=="red")/length(sample(jar, 10, replace=TRUE))) #Percent of marbles that are red out of the entire sample
## [1] "30%"

#7. Do the sampling 20 times, using the 'replicate' command. This should generate a list of 20 numbers. Each number is the mean of how many reds there were in 10 samples. Use your printVecInfo to see information of the samples. Also generate a histogram of the samples.

#List of 20 numbers each of which represents how many reds there were in our n=10 sample of jar
replicate(20, sum(sample(jar, size = 10, replace=TRUE) == "red"),simplify=TRUE)
##  [1] 3 4 5 7 9 3 5 7 8 7 6 5 5 6 8 3 1 7 5 5

#Descriptive statistics on the 20 repetitions in previous step (slightly different due to randomness)
printVecInfo(replicate(20, sum(sample(jar, size = 10, replace=TRUE) == "red"),simplify=TRUE))
## [1] 5.05
## [1] 5
## [1] 3
## [1] 8
## [1] 1.731291
##   5%  95%
## 3.00 7.05
## [1] 0.1087924

#Histogram visualization on the 20 repetitions in the previous step (slightly different due to randomness)
hist(replicate(20, sum(sample(jar, size = 10, replace=TRUE) == "red"),simplify=TRUE))
```

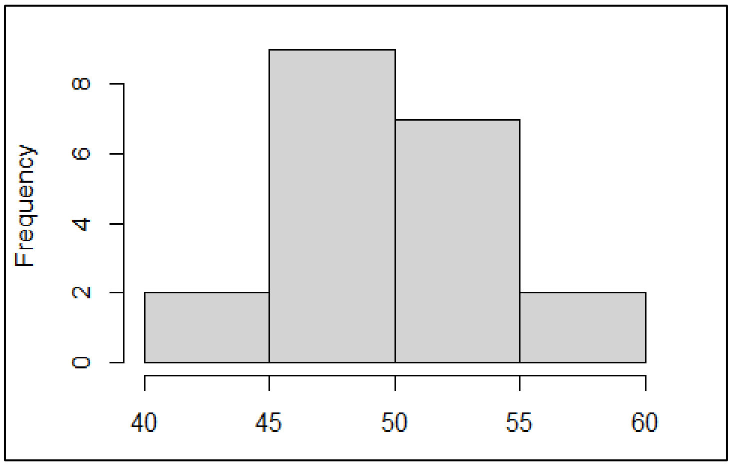


```r
#8. Repeat #7, but th                                                this time each number
represents the mean                                                  intVecInfo to see
information of the samples. Also generate a histogram of the samples.
```

```
#List of 20 numbers each of which represents how many reds there were in our n=100 sample of jar
replicate(20, sum(sample(jar, size = 100, replace=TRUE) == "red"),simplify=TRUE)
##  [1] 39 55 54 39 41 44 43 46 49 58 59 46 50 57 56 56 45 49 53 59

#Descriptive statistics on the 20 repetitions in previous step (slightly different due to randomness)
printVecInfo(replicate(20, sum(sample(jar, size = 100, replace=TRUE) == "red"),simplify=TRUE))
## [1] 50.7
## [1] 51
## [1] 42
## [1] 59
## [1] 5.07937
##    5%   95%
## 42.00 58.05
## [1] -0.2135598

#Histogram visualization on the 20 repetitions in the previous step (slightly different due to randomness)
hist(replicate(20, sum(sample(jar, size = 100, replace=TRUE) == "red"),simplify=TRUE))
```

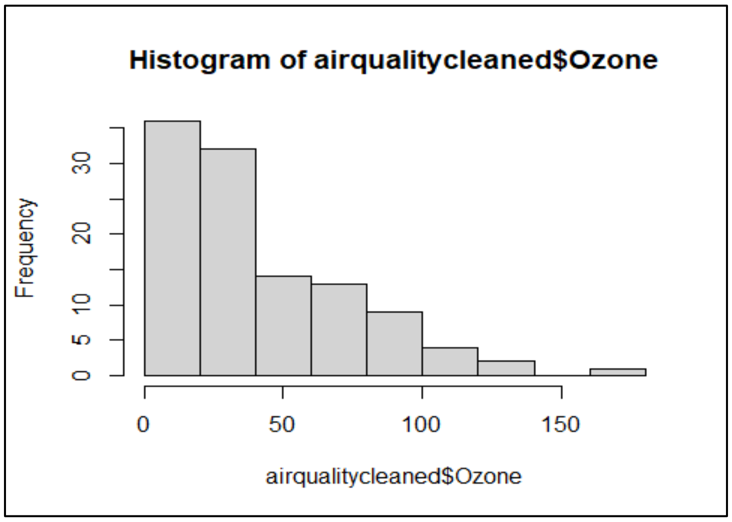


```
#9. Repeat #8, but this time, replicate the sampling 100 times. You should get 100 numbers, this time each
number represents the mean of how many reds there were in the 100 samples. Use your printVecInfo to
see information of the samples. Also generate a histogram of the samples.

#List of 100 numbers each of which represents how many reds there were in our n=100 sample of jar
replicate(20, sum(sample(jar, size = 100, replace=TRUE) == "red"),simplify=TRUE)
##  [1] 52 55 47 47 48 41 48 54 48 57 49 46 54 44 48 41 48 54 45 50

#Descriptive statistics on the 100 repetitions in previous step (slightly different due to randomness)
printVecInfo(replicate(20, sum(sample(jar, size = 100, replace=TRUE) == "red"),simplify=TRUE))
## [1] 49.85
## [1] 50.5
## [1] 43
## [1] 58
## [1] 4.533791
##    5%   95%
## 43.00 57.05
## [1] 0.04798539

#Histogram visualization on the 20 repetitions in the previous step (slightly different due to randomness)
hist(replicate(20, sum(sample(jar, size = 100, replace=TRUE) == "red"),simplify=TRUE))
```

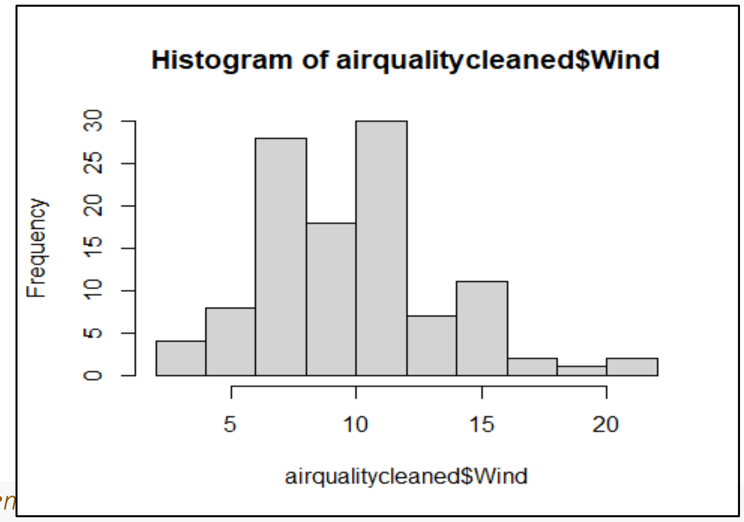


Step 3: Explore the airquality dataset

```
#10. Store the 'airquality' dataset into a temporary variable
airquality <- airquality

#11. clean the dataset (i.e. remove the NAs)
airqualitycleaned <- na.omit(airquality)

#12. Explore Ozone, Wind and Temp by doing a 'prntVecInfo' on each as well as generating a histogram fo
r each

#printVecInfo on Ozone and histogram
printVecInfo(airqualitycleaned$Ozone)

## [1] 42.0991
## [1] 31
## [1] 1
## [1] 168
## [1] 33.27597
##    5%   95%
##   8.5 109.0
## [1] 1.248104

hist(airqualitycleaned$Ozone)
```

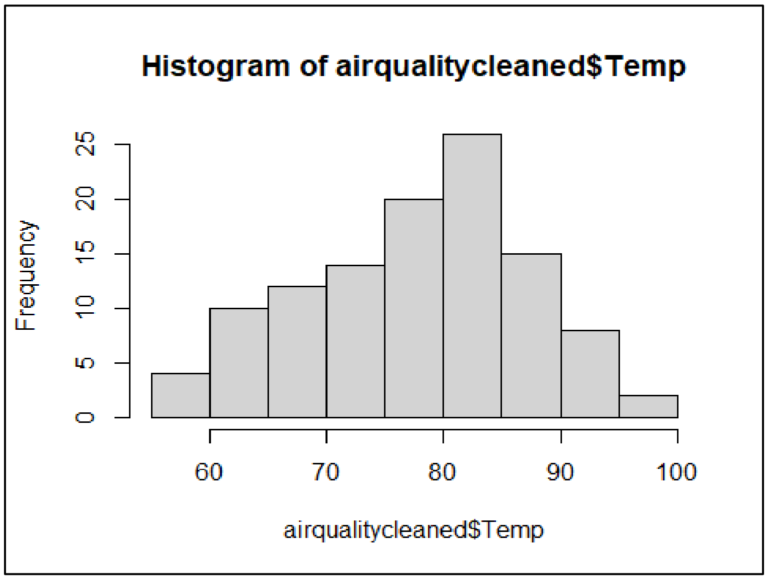


```
#printVecInfo on Wind and histogram
printVecInfo(airqualitycleaned$Wind)

## [1] 9.93964
## [1] 9.7
## [1] 2.3
## [1] 20.7
## [1] 3.557713
##   5%  95%
##  4.6 15.5
## [1] 0.4556414

hist(airqualitycleaned$Wind)
```



```
#printVecInfo on Ten
printVecInfo(airqualitycleaned$Temp)

## [1] 77.79279
## [1] 79
## [1] 57
## [1] 97
## [1] 9.529969
##   5%  95%
## 61.0 92.5
## [1] -0.2250959
```

```r
hist(airqualitycleaned$Temp)
```


Histogram of airqualitycleaned$Temp
Frequency
airqualitycleaned$Temp