CMTH 642 Data Analytics: Advanced Methods
Assignment 1

1. Read the csv files in the folder. (4 points)

```
loc = getwd()
macro = read.csv(paste(loc, "USDA_Macronutrients.csv", sep = "/"))
micro = read.csv(paste(loc, "USDA_Micronutrients.csv", sep = "/"))
```

2. Merge the data frames using the variable "ID". Name the Merged Data Frame "USDA". (4 points)

```
USDA = merge(macro, micro)
```

3. Check the datatypes of the attributes. Delete the commas in the Sodium and Potassium records. Assign Sodium and Potassium as numeric data types. (6 points)

```
sapply(USDA, class)
```

```
##           ID  Description     Calories      Protein     TotalFat
##    "integer"     "factor"    "integer"    "numeric"    "numeric"
## Carbohydrate       Sodium  Cholesterol        Sugar      Calcium
##    "numeric"     "factor"    "integer"    "numeric"    "integer"
##         Iron    Potassium     VitaminC     VitaminE     VitaminD
##    "numeric"     "factor"    "numeric"    "numeric"    "numeric"
```

```
USDA$Sodium = gsub(",", "", USDA$Sodium)
USDA$Potassium = gsub(",", "", USDA$Potassium)
USDA$Sodium = as.numeric(USDA$Sodium)
USDA$Potassium = as.numeric(USDA$Potassium)
```

4. Remove records (rows) with missing values in more than 4 attributes (columns). How many records remain in the data frame? (6 points)

```
na_count = apply(is.na(USDA), 1, sum)
USDA = USDA[na_count < 5,]
cat("Number of remaining records:", nrow(USDA))
```

```
## Number of remaining records: 6887
```

5. For records with missing values for Sugar, Vitamin E and Vitamin D, replace missing values with mean value for the respective variable. (6 points)

```
USDA$Sugar[is.na(USDA$Sugar)] = mean(USDA$Sugar[!is.na(USDA$Sugar)])
USDA$VitaminE[is.na(USDA$VitaminE)] =
mean(USDA$VitaminE[!is.na(USDA$VitaminE)])
USDA$VitaminD[is.na(USDA$VitaminD)] =
mean(USDA$VitaminD[!is.na(USDA$VitaminD)])
```

6. With a single line of code, remove all remaining records with missing values. Name the new Data Frame “USDAclean”. How many records remain in the data frame? (6 points)

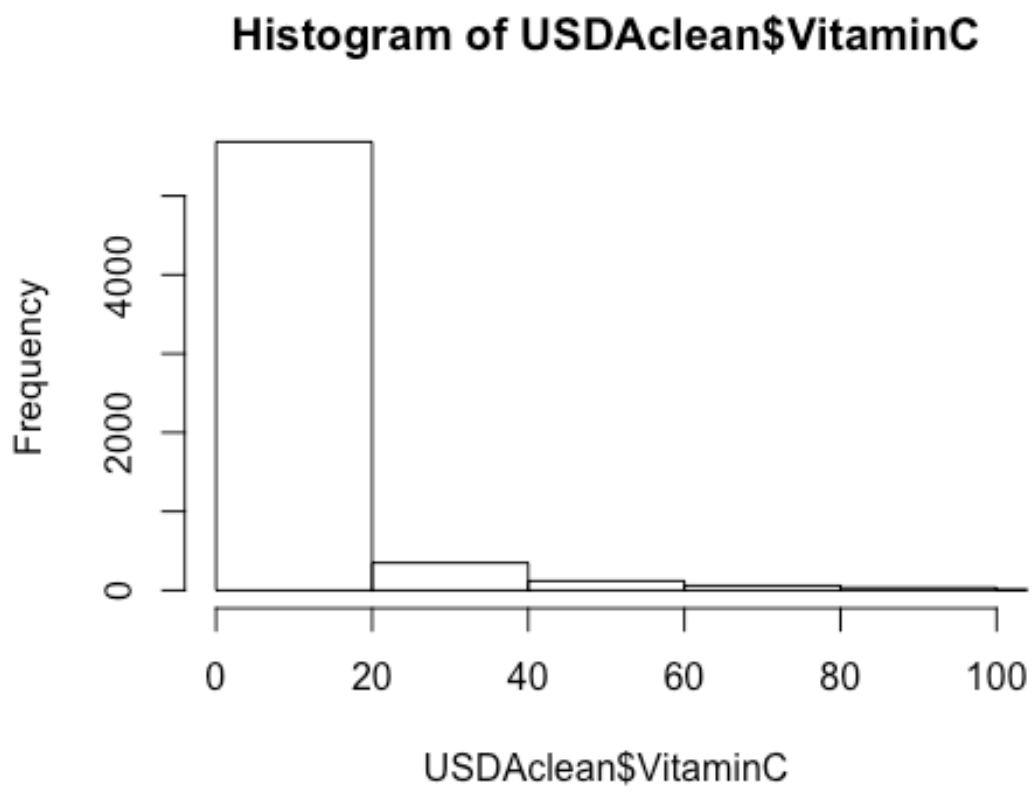
```
USDAclean = USDA[complete.cases(USDA),]
cat("Number of remaining records:", nrow(USDAclean))

## Number of remaining records: 6310
```

7. Which food has the highest sodium level? (6 points)

```
as.character(USDAclean$Description[USDAclean$Sodium ==
max(USDAclean$Sodium)])

## [1] "SALT,TABLE"
```

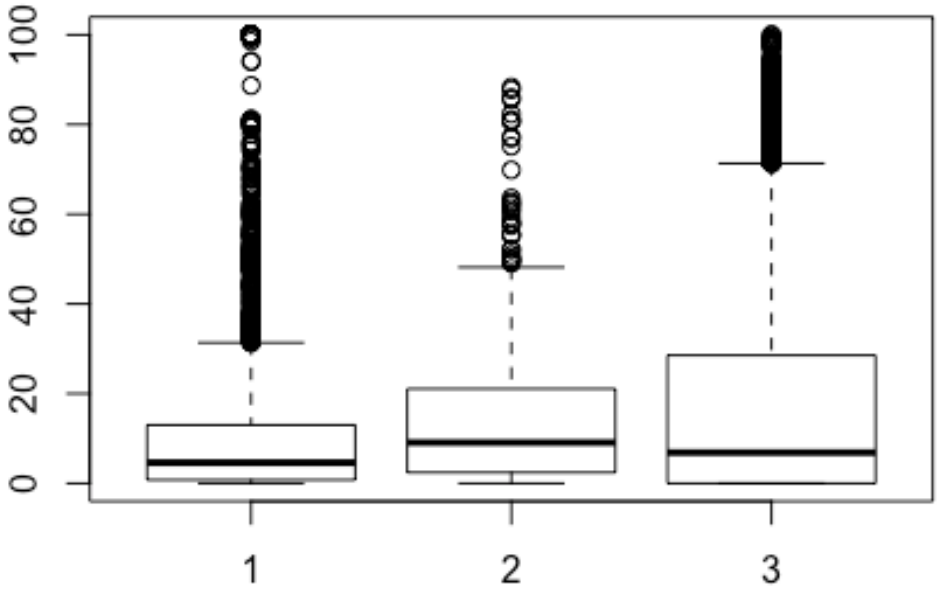
8. Create a histogram of Vitamin C distribution in foods, with a limit of 0 to 100 on the x-axis and breaks of 100. (6 points)

```
hist(USDAclean$VitaminC, xlim = c(0, 100), breaks = 100)
```



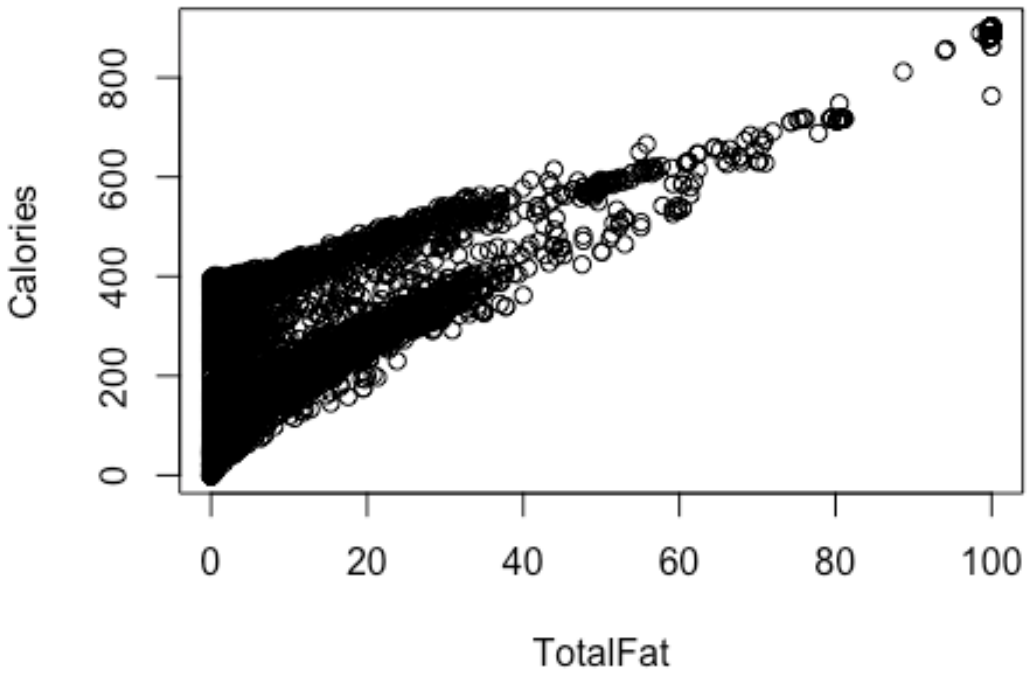
9. Create a boxplot to illustrate the distribution of values for TotalFat, Protein and Carbohydrate. (6 points)

```
with(USDAclean, boxplot(TotalFat, Protein, Carbohydrate))
```



10. Create a scatterplot to illustrate the relationship between a food's TotalFat content and its calorie content. (6 points)

```
with(USDAclean, plot(TotalFat, Calories))
```



11. Add a variable to the data frame that takes value 1 if the food has higher sodium than average, 0 otherwise. Call this variable HighSodium. Do the same for High Calories, High Protein, High Sugar, and High Fat. How many foods have both high sodium and high fat? (8 points)

```
USDAclean$HighSodium = 0
USDAclean$HighSodium[USDAclean$Sodium > mean(USDAclean$Sodium)] = 1

USDAclean$HighCalories = 0
USDAclean$HighCalories[USDAclean$Calories > mean(USDAclean$Calories)] =
1

USDAclean$HighProtein = 0
USDAclean$HighProtein[USDAclean$Protein > mean(USDAclean$Protein)] = 1

USDAclean$HighSugar = 0
USDAclean$HighSugar[USDAclean$Sugar > mean(USDAclean$Sugar)] = 1

USDAclean$HighFat = 0
USDAclean$HighFat[USDAclean$TotalFat > mean(USDAclean$TotalFat)] = 1
```

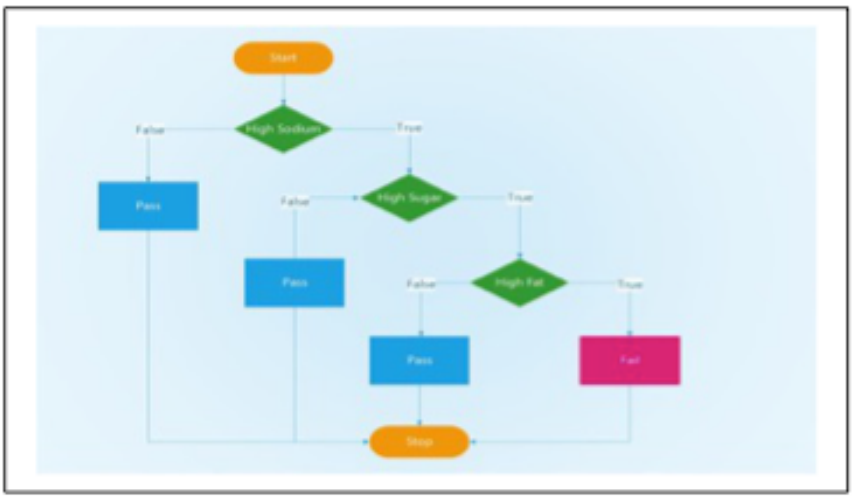
```
cat(sum(apply(USDAclean[c("HighSodium", "HighFat")], 1, function(x)
sum(x) == 2)), "foods have both high sodium and high fat.")

## 644 foods have both high sodium and high fat.
```

12. Calculate the average amount of iron, sorted by high and low protein. (8 points)

```
with(USDAclean, tapply(Iron, HighProtein, mean))

##        0        1
## 2.696634 3.069541
```

13. Create a script for a "HealthCheck" program to detect unhealthy foods. Use the algorithm flowchart below as a basis for this script. (8 points)

```
#install.packages('jpeg')
require(jpeg)

## Loading required package: jpeg

img <- readJPEG("HealthCheck.jpg")
plot(1:4, ty = 'n', ann = F, xaxt = 'n', yaxt = 'n')
rasterImage(img,1,1,4,4)
```



```r
healthcheck = function(x) {
  if (x$HighSodium == 0) return("Pass")
  else if (x$HighSugar == 0) return("Pass")
  else if (x$HighFat == 0) return("Pass")
  else return("Fail")
}
```

14. Add a new variable called HealthCheck to the data frame using the output of the function. (8 points)

```r
for (i in 1:nrow(USDAclean)) {
  USDAclean$HealthCheck[i] = healthcheck(USDAclean[i,])
}
```

15. How many foods in the USDAclean data frame fail the HealthCheck? (8 points)

```r
sum(USDAclean$HealthCheck == 'Fail')
```

```
## [1] 237
```

16. Save your final data frame as “USDAclean_ [your last name]” (4 points)

```r
write.csv(USDAclean, "USDAclean_Dagys")
```