Homework 01 R Basics

Due by 11:59pm, Saturday, 1.27.24

S&DS 230

This homework assignment contains 5 problems. Edit this `.Rmd` file and insert your responses under the appropriate problems. Submit the knitted output of this file either as a .pdf or .doc file. To make a pdf file, knit to .doc or .html and then convert to .pdf in either Word or your browser.

Our intent is to help you

1. Practice basic R coding
2. Practice basic R markdown
3. Start dealing with data

Don't panic, don't be surprised if you're frustrated. Come to office hours with JDRS or our excellent TAs/ULAs. You can also post questions through ED Discussion on CANVAS. You are welcome to work with each other - however, you **MUST** submit your own assignment that represents your own solutions.

Before submitting, delete the instructions on lines 17 through 28

(1) RMarkdown Practice *(24 points)* Change the markdown code below as indicated.

Make this line bold

Make this line italics

###Make this line a third level header###

- Make this line a bullet point
  - Make this line an indented (or level two) bullet point

LINK (make the word LINK at left link to the New York Times home page AND make it bold)

`Make this line look like R Code`

Below this line, insert a new R chunk, create a vector called `xvec` that contains the integers 2 through 7, and have R display what is in `xvec`.

```
xvec <- 2:7
print(xvec)
```

```
## [1] 2 3 4 5 6 7
```

(2) R Syntax Practice *(12 points)* Modify the R code below to follow good R Syntax practices

```
x <- 5

x <- c(1, 2, 3)

length(x)

for (i in 1:10){
  x  <- 1 + 1
```

```
}

x <- 1
y <- c(3, 4)
```

(3) Data handling *36 pts*

(3.1) Insert a new R code chunk below.
(3.2) Read the .csv stored HERE into a new data frame and call is "wb". This is the World Bank data I discussed in class two.

(3.3) Get the dimension of wb.

(3.4) Get the variable names of wb.

(3.5) Show the first 6 lines of wb.

(3.6) Get the data type of each variable.

(3.7) What is the data type of the variable `Pop`?

(3.8) Create a new object called `subset` that has only the variables Country, GNI, Exports, and Imports AND only for countries where GNI is greater than 70000. You'll need to use the `na.omit()` function (use `help(na.omit)`) to eliminate countries missing data for any of the four variables you retain. You should end up with exactly three countries in `subset`.

(3.9) Get summary statistics for cell phone lines per 100 people (called `Cell`). The function you want is `summary()`.

(3.10) Store the results from (3.9) in a new object called `stats`. Incidentally, `stats` will be a vector!

(3.11) Get the length of `stats`. The function you want is `length()`.

(3.12) Get r to show the following elements of `stats` : 1,2,3,5,6

```
#3.2
wb <- read.csv("http://reuningscherer.net/S&DS230/data/WB.2016.csv")

#3.3
dim(wb)
```

```
## [1] 217  29
```

```
#3.4
names(wb)
```

```
##  [1] "Country"       "Code"          "Population"    "Rural"
##  [5] "GNI"           "IncomeTop10"   "Imports"       "Exports"
##  [9] "Military"      "Cell"          "Fertility66"   "Fertility16"
## [13] "Measles"       "InfMort"       "LifeExp"       "PM2.5"
## [17] "Diesel"        "CO2"           "EnergyUse"     "FossilPct"
## [21] "Forest94"      "Forest14"      "Deforestation" "GunTotal"
## [25] "GunHomicide"   "GunSuicide"    "GunUnint"      "GunUndet"
## [29] "GunsPer100"
```

```
#3.5
head(wb)
```

```
##          Country Code Population  Rural  GNI IncomeTop10  Imports  Exports
## 1    Afghanistan  AFG   34656032 72.868  580          NA 49.02498  6.89625
## 2        Albania  ALB    2876101 41.624 4320          NA 45.74585 28.92342
## 3        Algeria  DZA   40606052 28.696 4360          NA 35.27028 21.00176
```

```
## 4 American Samoa  ASM      55599 12.852   NA          NA 93.46505 65.04559
## 5        Andorra  AND      77281 15.388   NA          NA       NA       NA
## 6         Angola  AGO   28813463 55.181 3450          NA 29.41717 30.01704
##   Military      Cell Fertility66 Fertility16 Measles InfMort LifeExp     PM2.5
## 1 0.954643  62.33542       7.450       4.635      62    53.2  63.673 62.854857
## 2 1.101507 115.15226       5.581       1.713      96    12.0  78.345 14.634008
## 3 6.424474 115.84805       7.676       2.776      94    21.6  76.078 37.230956
## 4       NA        NA          NA          NA      NA      NA      NA  3.763412
## 5       NA  92.04332          NA          NA      97     2.4      NA 10.879472
## 6 2.962392  45.12170       7.618       5.694      49    54.6  61.547 36.240479
##   Diesel      CO2 EnergyUse FossilPct Forest94 Forest14 Deforestation GunTotal
## 1   0.70 0.299445        NA        NA   336198   274088   18.47423245       NA
## 2   1.35 1.978763  808.4558  61.42180  1286610  1244430   3.278382727       NA
## 3   0.17 3.717410 1321.0995  99.97792  5365326  4945220   7.830018157       NA
## 4     NA       NA        NA        NA  1650846  2067791  -25.25647462       NA
## 5     NA 5.832170        NA        NA   147772   168760  -14.20296132       NA
## 6   0.82 1.291328  545.0405  48.27955   113216   114170  -0.842637083       NA
##   GunHomicide GunSuicide GunUnint GunUndet GunsPer100
## 1          NA         NA       NA       NA         NA
## 2          NA         NA       NA       NA         NA
## 3          NA         NA       NA       NA         NA
## 4          NA         NA       NA       NA         NA
## 5          NA         NA       NA       NA         NA
## 6          NA         NA       NA       NA         NA
```

```
#3.6
sapply(wb, class)
```

```
##       Country          Code    Population         Rural           GNI
##   "character"   "character"     "integer"     "numeric"     "integer"
##   IncomeTop10       Imports       Exports      Military          Cell
##     "numeric"     "numeric"     "numeric"     "numeric"     "numeric"
##   Fertility66   Fertility16       Measles       InfMort       LifeExp
##     "numeric"     "numeric"     "integer"     "numeric"     "numeric"
##         PM2.5        Diesel           CO2     EnergyUse     FossilPct
##     "numeric"     "numeric"     "numeric"     "numeric"     "numeric"
##      Forest94      Forest14 Deforestation      GunTotal   GunHomicide
##     "numeric"     "numeric"   "character"     "numeric"     "numeric"
##    GunSuicide      GunUnint      GunUndet    GunsPer100
##     "numeric"     "numeric"     "numeric"     "numeric"
```

```
#3.7
typeof(wb$Pop)
```

```
## [1] "integer"
```

```
#3.8
na.omit(wb[, c("Country", "GNI", "Exports", "Imports")][wb$GNI > 70000, ])
```

```
##         Country   GNI   Exports   Imports
## 116  Luxembourg 71590 221.26778 186.16333
## 147      Norway 82010  34.13664  33.27319
## 189 Switzerland 82080  65.81131  54.58890
```

```
#3.9
summary(wb$Cell)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   10.21   81.69  110.66  106.78  127.97  321.80      17
```

```
#3.10
stats <- as.vector(summary(wb$Cell))
print(stats)
```

```
## [1]  10.21264  81.68643 110.66193 106.78417 127.97427 321.80304  17.00000
```

```
#3.11
length(stats)
```

```
## [1] 7
```

```
#3.12
stats[c(1, 2, 3, 5, 6)]
```

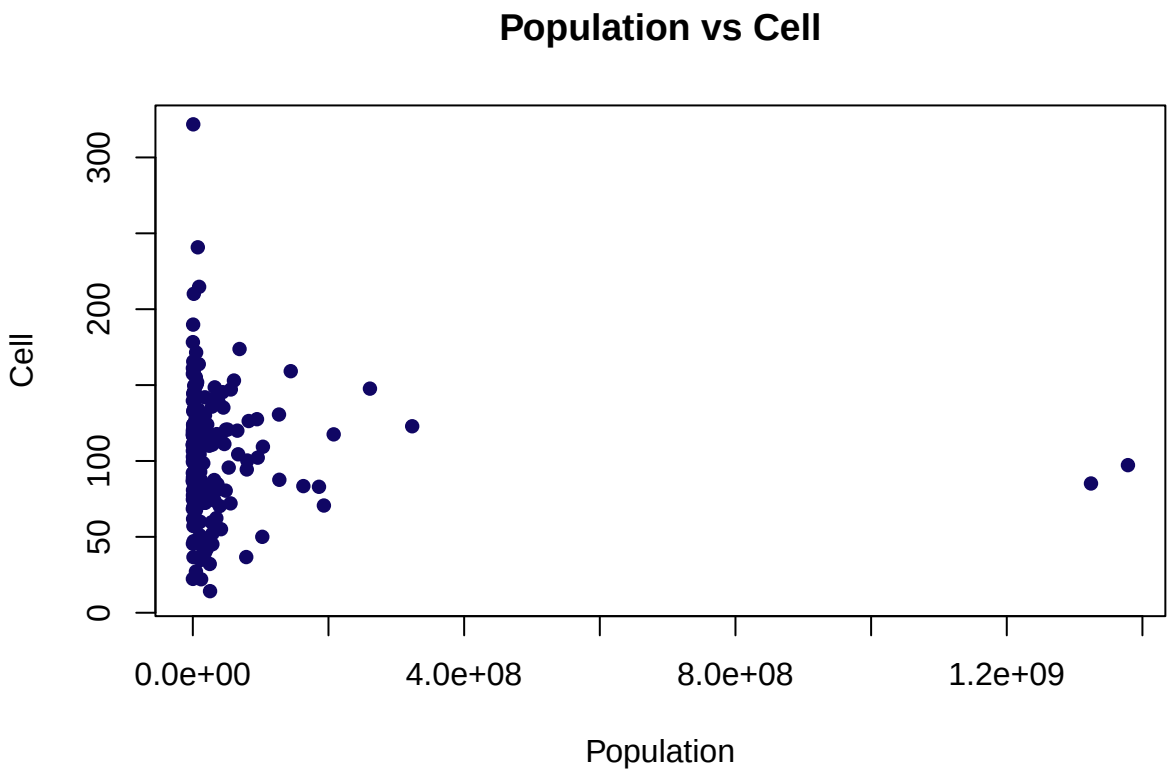
```
## [1]  10.21264  81.68643 110.66193 127.97427 321.80304
```

(4) Plots *16 pts*

(4.1) Using the `wb` dataset created above, make a scatterplot of "Population" on the x axis and "Cell" on the y axis. Include a main title, axis titles, and a non-default symbol color and symbol type. *Hint: check out ?par or see examples from class 1 or class 3 R code.*

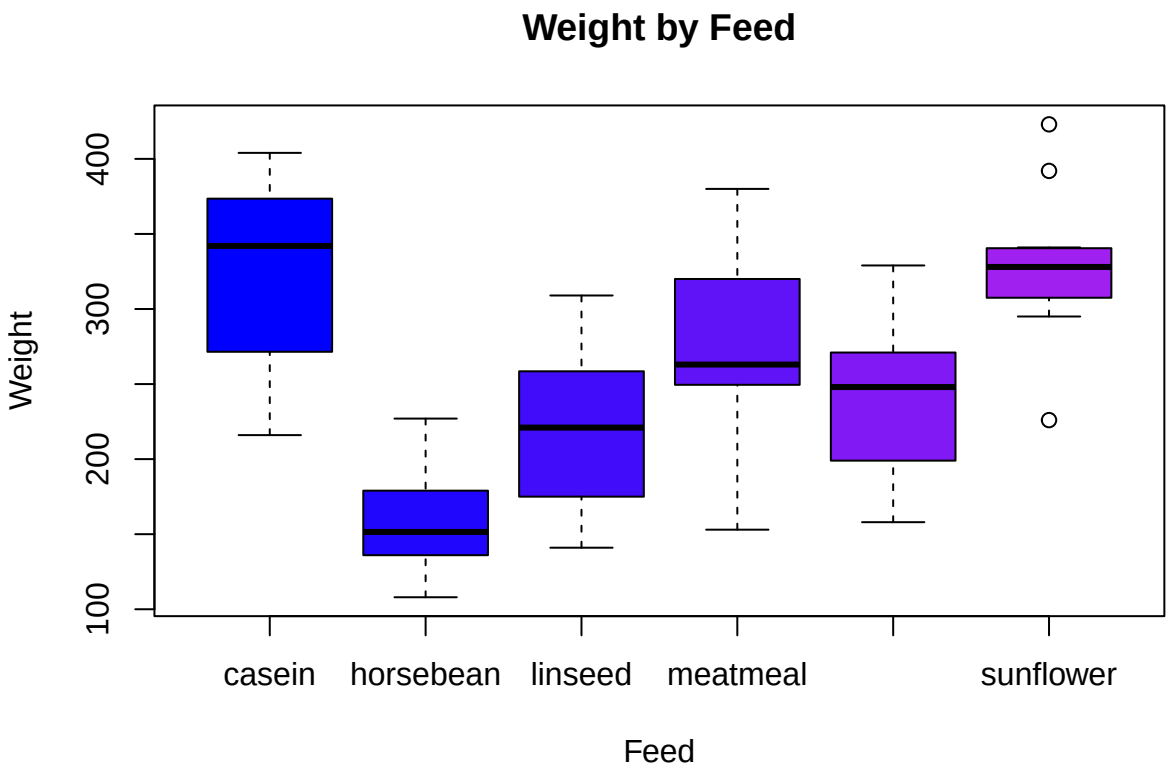
(4.2) Use the `data()` function to load the "chickwts" dataset that comes with base R's "datasets" package. Then, create a boxplot of chicken weight by feed type. Ensure the plot has a main title, axis labels, and a unique color for each feed type. You can learn about the dataset by typing `?chickwts`.

```
#4.1
plot(wb$Population, wb$Cell, main = "Population vs Cell",
     xlab = "Population", ylab = "Cell", col = "#100664", pch = 16)
```




```r
#4.2
data(chickwts)
gradient <- colorRampPalette(c("blue", "purple"))

boxplot(weight ~ feed, data = chickwts,
        main = "Weight by Feed",
        xlab = "Feed", ylab = "Weight",
        col = gradient(length(unique(chickwts$feed))))
```



(5) Lists *12 pts* The code below creates a list called `aList`

(5.1) Compute the sum of the second element of the list's third element. Store the result into an object named `mySum`. You'll want to use the `sum()` function.

```
aList <- list(c(1, 5, 4), letters[c(1, 6, 4, 9, 22, 3)], list(c(1, 1, 1),
                        c(14, 13, 12), c(3, 2, 1)), c(runif(8)))
mySum <- sum(aList[[3]][[2]])
print(mySum)
```

```
## [1] 39
```

(5.2) What is the difference between what is returned from the following two commands?

```
aList[[3]][2]
```

```
## [[1]]
## [1] 14 13 12
```

```
aList[[3]][[2]]
```

```
## [1] 14 13 12
```

The first command first gets a sublist of the first 3 elements of aList and then gets the 2nd element of the sublist because the way the sublist is returned with labeled indexes, we also get a doulbe nested 1 before The seccond command directly gets the second element of the third element of aList, so it just returns that