MA 354: Data Analysis I – Fall 2019
Homework 0:

Complete the following opportunities to use what we've talked about in class. These questions will be graded for correctness, communication and succinctness. Ensure you show your work and explain your logic in a legible and refined submission. If you make a mistake while completing this assignment do not delete it – instead, make a note about why it's a mistake, how you realized it, and how you fixed it. The best, most productive recovery from a mistake will receive a 5 point bonus.

0. **Complete weekly diagnostics.** Upon completion of this assignment reflect, in a few sentences, on what you've learned since the beginning of the semester. Think about an mention any concepts you want or need to continue working on and list at least one goal for the next assignment.

1. **Writing Functions and Loops.** Create a function called `f1` in `R`, in as few lines as possible that represents the following function.

$$f_1(x) = \ln\left(\sqrt{x * 10^3}\right)$$

   Compare the runtime of your function to the one below using a loop that saves the runtime of each function over 1000 iterations. Summarize and compare the results.

```
> f2 <- function(x){
+   a<- x*10^3
+   b<- sqrt(a)
+   c<- log(b)
+   return(c)
+ }
> f2(1)
```

```
[1] 3.453878
```

   You can track the runtime of code using the following `R` code.

```
> start_time <- Sys.time()
> #do things you want to time here
> end_time <- Sys.time()
> runtime <-end_time - start_time
```

   You might want to report this runtime in your .pdf document, but you'll find that the runtime changes each time you compile your document. You can pass variable values in `R` to the document using the `Sexpr` command. For example, the runtime of the comment above is 0.001 seconds.

2. **Conditional Statements and Loops.** Assess which is faster.

   (a) Generate 1000 random integers between 1 and 1000 with replacement with the following `R` code.

```
> random.ints<-sample(x=1:1000,size=1000,replace=TRUE)
```

   Then, using a loop, create a vector named `random.type` that is "odd" when the corresponding value in `random.ints` is odd and "even" otherwise.

   i. Add elements to the vector called `random.type` as they are decided; e.g., for an iteration where the value is odd we add an element using `random.type<-c(random.type,"odd")`.
   ii. Create an empty vector of size 1000 (with `NA` entries) to start and place elements into the vector; e.g., for iteration i where the value is odd we place the element using `random.type[i]`.

   Which is faster?

3. **Conditional Statements and Loops** Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting with 1 and 2, the first 10 terms will be 1, 2, 3, 5, 8, 13, 21, 34, 55, 89. Find the sum of the even terms in the Fibonacci sequence whose values do not exceed one million.

4. **Conditional Statements and Loops** A palindromic number reads the same both ways. The largest palindrome made from the product of two 2-digit numbers is $9009 = 91 \times 99$. Find the largest palindrome made from the product of two 3-digit numbers.

   You can get the reverse of a number using the "stringr" package for R (practice Sweave by putting by the citation here) as follows.

```
> install.packages("stringr",repos = "https://cloud.r-project.org")

The downloaded binary packages are in
        /var/folders/hd/yj9qx9890x195xhg0xyq4xyh0000gq/T//Rtmpolkfbf/downloaded_packages

> library(stringr)
> number.to.reverse <- 1234
> #creates a list containing a vector of characters
> (split.number<-str_split(number.to.reverse,pattern=""))

[[1]]
[1] "1" "2" "3" "4"

> (split.number<-split.number[[1]]) #takes just the vector

[1] "1" "2" "3" "4"

> (split.number.reversed <- rev(split.number)) #reverses the order of the vector

[1] "4" "3" "2" "1"

> (reversed.number <- paste(split.number.reversed, collapse="")) #paste items back together

[1] "4321"

> (reversed.number<-as.numeric(reversed.number)) #treat it like a number

[1] 4321
```

   Remark: You'll find that the solution to this question will elongate the compiling time for the .pdf. That's okay. Once you're happy with the solution you can type the output into your code chunk and set eval=FALSE between the left and right arrows.

5. **(Working with Data)** Below you will load and summarize a dataset containing 575 observations of drug treatments. The data includes the following

   - ID – Identification Code (1 - 575)
   - AGE – Age at Enrollment (Years)
   - BECK – Beck Depression Score (0.000 - 54.000)
   - HC – Heroin/Cocaine Use During 3 Months Prior to Admission (1 = Heroin & Cocaine; 2 = Heroin Only, 3 = Cocaine Only; 4 = Neither Heroin nor Cocaine)
   - IV – History of IV Drug Use (1 = Never; 2 = Previous; 3 = Recent)
   - IV3 – Recent IV use (1 = Yes; 0 = No)
   - NDT – Number of Prior Drug Treatments (0 - 40)
   - RACE – Subject's Race (0 = White; 1 = Non-White)
   - TREAT – Treatment Randomization (0 = Short Assignment; 1 = Long Assignment)
   - SITE – Treatment Site (0 = A; 1 = B)
   - LEN.T – Length of Stay in Treatment (Days Admission Date to Exit Date)
   - TIME – Time to Drug Relapse (Days Measured from Admission Date)
   - CENSOR – Event for Treating Lost to Follow-Up as Returned to Drugs (1 = Returned to Drugs or Lost to Follow-Up; 0 = Otherwise)
   - etc.

   (a) Load the data provided in the "quantreg" package for R (Koenker, 2018).

   ```
   > install.packages("quantreg",repos = "http://cloud.r-project.org/")
   > library("quantreg")
   > data("uis")
   ```

   (b) Numerically summarize the Beck Depression Score of the observed drug treatment patients. Note the following designations of the test when interpreting your results.

   - 0-13: minimal depression
   - 14-19: mild depression
   - 20-28: moderate depression
   - 29-63: severe depression.

   (c) Graphically summarize the Beck Depression Score of the observed drug treatment patients. Interpret the results through the lens of the scale above.

   (d) List three questions about drug use that we might be able to answer based on the data we have.

6. **(Working with Data)** Hepatitis C is a disease that affects the liver. The virus that causes hepatitis C is spread through blood or bodily fluids of an infected person. The virus is often difficult to diagnose because there are few unique symptoms. Those infected, however, sometimes experience jaundice – a condition that causes yellowing of the skin or eyes, as the liver is infected.

   Bracht et al. (2016) consider the human microfibrillar-associated protein 4, or MFAP4, and its role in disease-related tissue. Stage 0–no fibrosis; Stage 1–enlarged, fibrotic portal tracts; Stage 2–periportal fibrosis or portal-portal septa, but intact architecture; Stage 3–fibrosis with architectural distortion, but no obvious cirrhosis; and Stage 4–probable or definite cirrhosis.

   Previously, it has been shown that MFAP4 is a biomarker candidate for hepatic fibrosis and cirrhosis in hepatitis C patients. The analysis of Bracht et al. (2016) aimed to consider the ability of MFAP4 to differentiate between stages of the disease – fibrosis stages (0-2) and cirrhosis (3-4) based on the Scheuer scoring system.

   Below, I load the data and calculate the age of patients using the "lubridate" package for R (Grolemund and Wickham, 2011).

```
> fn<-"http://cipolli.com/students/data/biomarker.csv"
> dat <- read.csv(file=fn, header=TRUE, sep=",")
> head(dat)

  Patient.ID Year.of.Birth Gender Date.of.sampling Fibrosis.Stage HCV.Genotype
1       1112          1958 female         2/1/2005              0            1
2       3403          1946 female        1/18/2005              2             
3       2841          1954 female         1/3/2005              3            1
4        654          1958   male         2/1/2005              3            1
5       2788          1960   male        12/9/2004              0            3
6       2242          1954 female        5/12/2004              0            1
  MFAP4.U.mL
1        5.1
2        5.3
3       12.9
4        6.2
5        3.3
6        7.5

> ###Calculate the age of each subject
> install.packages("lubridate",repos = "http://cloud.r-project.org/")

The downloaded binary packages are in
        /var/folders/hd/yj9qx9890x195xhg0xyq4xyh0000gq/T//Rtmpolkfbf/downloaded_packages

> library(lubridate)
> ###Create Date Variable for Date Sampled
> dos<-mdy(dat$Date.of.sampling)
> dos.year<-year(dos)
> ###Create age Variable
> age <- dos.year - dat$Year.of.Birth
> ###Add age to original dataset
> dat<-data.frame(dat,age)
```

   (a) Recreate Table 1 in https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4932744/. Add a row to this table labeled "Median age IQR." I've provided the LaTeX code for the table and made the first entry so you can see how it works.

Fibrosis Stage	F0	F1	F2	F3	F4	All
Age						
Mean age SD	43.75 ± 12.11					
Median age IQR						
Gender						
Women						
Men						
Number of patients						
HCV genotype						
1						
2						
3						
4						
Other						
NA						

Table 1: Patient cohorts characteristics, subdivided by fibrosis stage

(b) Create several graphs that may be helpful for the researchers.

References

Bracht, T., Molleken, C., Ahrens, M., Poschmann, G., Schlosser, A., Eisenacher, M., Stuhler, K., Meyer, H. E., Schmiegel, W. H., Holmskov, U., Sorensen, G. L., and Sitek, B. (2016). Evaluation of the biomarker candidate mfap4 for non-invasive assessment of hepatic fibrosis in hepatitis C patients. *Journal of Translational Medicine*, 14.

Grolemund, G. and Wickham, H. (2011). Dates and times made easy with lubridate. *Journal of Statistical Software*, 40(3):1–25.

Koenker, R. (2018). *quantreg: Quantile Regression*. R package version 5.35.