# Module 5 Lab: advanced data manipulation

*BIOSTAT 213: Introduction to Computing in the R Software Environment*

*Wednesday, February 13, 2019*

Save your code in a script file. I suggest adding labels and notes to this script file as comments. You may work in groups of 2–3.

1. *Use* ***presidencies.csv****.* If you have not already, convert `start`, `end`, and `dob` to Date objects.
2. *Use* ***presidencies.csv****.* Find the president who was youngest at the start of his term. How old was he? Find the president who was oldest at the start of his term. How old was he?
3. *Use* ***presidencies.csv****.* Find the president who was youngest at the end of his term. How old was he? Find the president who was oldest at the end of his term. How old was he?
4. *Use* ***presidencies.csv****.* Find the president with the longest term. How long was that term? Find the president with the shortest term. How long was that term?
5. *Use* ***kevin_love.txt****.* Reshape the data to long format, so that each row represents a season-statistic. The following statistics should be included: 2-point percent (`fg2p`), 3-point percent (`fg3p`), and free-throw percentage (`ftp`). Your data should look like this:

```
> head(mm[order(mm$season),])
                                season age                    stat value
2008-09.Two-point field goals   2008-09  20   Two-point field goals 0.469
2008-09.Three-point field goals 2008-09  20 Three-point field goals 0.105
2008-09.Free throws             2008-09  20             Free throws 0.789
2009-10.Two-point field goals   2009-10  21   Two-point field goals 0.474
2009-10.Three-point field goals 2009-10  21 Three-point field goals 0.330
2009-10.Free throws             2009-10  21             Free throws 0.815
```

6. *Use* ***DEMO_H.XPT and DR1TOT_H.*** Download and import the 2 data files.
   (a) Subset the demographic data set to individuals 50 years of age or younger. Use this reduced demographic data set for the subsequent problems.
   (b) Check for duplicated ID (`SEQN`) in each data set.
   (c) Merge the dietary data into the demographic data by ID (`SEQN`), using `all=TRUE`, and giving the new data frame a new arbitrary name. Check the number of rows. This should match the following, since `all=TRUE` means *X OR Y*:

```
> length(unique(c(tt1$seqn,tt2$seqn)))
[1] 10100
```

   (a) Merge the dietary data into the demographic data by ID (`SEQN`), using `all=FALSE`, and giving the new data frame a new arbitrary name. Check the number of rows. This should match the following, since `all=FALSE` means *X AND Y*:

```
> sum(is.element(tt1$seqn,tt2$seqn))
[1] 7213
> sum(is.element(tt2$seqn,tt1$seqn))
[1] 7213
```

   (a) Merge the dietary data into the demographic data by ID (`SEQN`), using `all.x=TRUE, all.y=FALSE`, and giving the new data frame a new arbitrary name. Check the number of rows. This should match the following, since `all.x=TRUE, all.y=FALSE` means simply *X*:

```
> length(unique(tt1$seqn))
[1] 7500
```

(a) Merge the dietary data into the demographic data by ID (`SEQN`), using `all.x=FALSE, all.y=TRUE`, and giving the new data frame a new arbitrary name. Check the number of rows. This should match the following, since `all.x=FALSE, all.y=TRUE` means simply $Y$:

```
> length(unique(tt2$seqn))
[1] 9813
```