Exercise: **Control Flow**

Day 4, Part A

```
> library(foreign)
> library(ggplot2)
> library(reshape2)
> library(microbenchmark)
```

1. There is a series of files containing different years of data on income and poverty in Washington state ("data/wa_income_[year]"). Load and combine all of the data in these files into a single data frame. Hint: this requires just one loop combined with some `if/else` statements.

```
> main_dir <- "C:/Users/ngraetz/Documents/repos/r_training_penn/" # CHANGE TO YOUR LOCAL COPY OF
>
> data <- NULL
> for (year in 1997:2015) {
+   if (year < 2004) {
+     sub <- read.csv(paste0(main_dir, "/data/wa_income_", year, ".csv"))
+   } else {
+     sub <- read.dta(paste0(main_dir, "/data/wa_income_", year, ".dta"))
+     sub <- plyr::rename(sub, c("FIPS" = "fips", "median_income" = "income_median"))
+   }
+   data <- rbind(data, sub)
+ }
> summary(data)
fips             year         income_median      poverty
 Min.   :53001   Min.   :1997   Min.   :27453   Min.   : 6.60
 1st Qu.:53019   1st Qu.:2001   1st Qu.:36992   1st Qu.:11.50
 Median :53039   Median :2006   Median :42369   Median :14.10
 Mean   :53039   Mean   :2006   Mean   :43726   Mean   :14.26
 3rd Qu.:53059   3rd Qu.:2011   3rd Qu.:48693   3rd Qu.:16.40
 Max.   :53077   Max.   :2015   Max.   :81816   Max.   :32.30
```

2. Make a line plot of median household income (y-axis) vs year (x-axis) for each county, saving these as separate pages in a PDF. Hint: the `unique()` function is useful for finding all the unique values of a vector.

```
> pdf(paste0(main_dir, "output/wa_median_income_trends.pdf"), width=10, height=8)
> for (cnty in unique(data$fips)) {
+   gg <- ggplot(data[data$fips == cnty,], aes(x=year, y=income_median)) +
+     geom_line() + labs(title = paste("County:", cnty))
+   print(gg)
+ }
> dev.off()
pdf
  2
```

3. Using a loop, calculate the mean poverty rate in each year. Do the same using `dcast()` and compare your results.

```
> mean_poverty1 <- data.frame(year = 1997:2015, poverty = NA)
> for (yy in 1997:2015) {
+   mean_poverty1[mean_poverty1$year == yy, "poverty"] <-
+     mean(data[data$year == yy, "income_median"])
+ }
>
> mean_poverty2 <- dcast(data, year ~ ., value.var = "income_median",
+                        fun.aggregate = mean)
> names(mean_poverty2)[2] <- "poverty"
>
> all.equal(mean_poverty1, mean_poverty2)
[1] TRUE
```

Bonus:

4. Using the `microbenchmark()` function in the `microbenchmark` library, determine which of the two approaches in question 3 is faster, and by how much. Which approach do you prefer? Is the difference in timing enough to sway your opinion? Hint: https://www.r-bloggers.com/5-ways-to-measure-running-time-of-r-code/ has some helpful examples of using `microbenchmark()`.

```
> microbenchmark("loop" = {
+                  mean_poverty1 <- data.frame(year = 1997:2015, poverty = NA)
+                  for (yy in 1997:2015) {
+                    mean_poverty1[mean_poverty1$year == yy, "poverty"] <-
+                      mean(data[data$year == yy, "income_median"])
+                  }
+                },
+                "dcast" = {
+                  mean_poverty2 <- dcast(data, year ~ ., value.var = "income_median",
+                                         fun.aggregate = mean)
+                  names(mean_poverty2)[2] <- "poverty"
+                })
Unit: milliseconds
  expr      min       lq     mean   median       uq       max neval cld
  loop 1.340535 1.550222 2.539191 1.973241 2.796307  7.589922   100   a
 dcast 1.646131 1.878245 2.897252 2.304728 3.075828 10.186388   100   a
```