FISH 552
Homework 3 Solutions

Question 1

a) Read in the three data sets.

```
colNames <- c("Year", "spawners", "recruits",
               "catch","fishMortality")

mack.black <- read.table("http:/. . ./MACKBLACK.dat",
                          na.strings = ".", col.names = colNames)

mack.nafo <- read.table("http:/. . ./MACK2-6.dat",
                         na.strings = ".", col.names = colNames)

mack.ices <- read.table("http:/. . ./MACKWS.dat",
                         na.strings = ".", col.names = colNames)
```

I did not given any specific instructions on how to read this data in except to conveniently name the columns. Lots of people forgot to specify the missing values as `NA`. Always remember to do this. R will be much happier. Otherwise those columns will be read in as a factor rather than numeric (because of the character data) with potentially bad side effects. You may have run into issues plotting this data, but having numeric data incorrectly read in as factors will also cause problems with statistical analyses.

b) Create a single data set from `mack.black` and `mack.nafo` that contains all of the variables in each data set (`spawners recruits catch fishMortality`), but is restricted to years that the two data sets have in common. One function will do this. Call this data frame `mack.partial`. The column names of `mack.partial` should have suffices corresponding to the specific data set. By that I mean: `"spawners.black" "recruits.black" . . . "spawners.nafo" "recruits.nafo" . .` Specifying one option in this function will do this for you.

```
mack.partial <- merge(mack.black, mack.nafo, by="Year",
                      suffixes=c(".black",".nafo"))
```

`merge` is such a cool function !

c) Create a single data set from `mack.partial` and `mack.ices` that contains all of the variables in each data set, but is restricted to years that the two data sets have in common. One function will do this. Call this data frame `mack`.

```
mack <- merge(mack.partial, mack.ices,  by="Year")
```

Note that the `suffixes` option would not work here because the columns of the merged data set are unique. Hence part d).

d) Change the variable names in `mack` that are uninformative to `"spawners.ices"` *"recruits.ices" "catch.ices" "fishMortality.ices".*

```
names(mack)[10:13] <- c("spawners.ices","recruits.ices",
                        "catch.ices","fishMortality.ices")
```

e) Create this graph with the data in `mack`. A slick way to do this is with the `matplot()` function. Look up the help on this function to get started. If you prefer to not use this function you may use `plot()` and then `lines()`.

```
matplot(mack[,1], mack[,c(2,6,10)],
             type = 'l', lty = 2,
             xlab = "Years", ylab = "Spawner Biomass",
             main = "Atlantic mackerel")
legend("topleft",lty = 2, col = 1:3,
       c("Black Sea","NAFO","ICES"), bty='n')
```

Question 2

For this question we'll be creating our own set of simulated data.

a) Create a data frame named `temperature` which has 2 columns: the dates Jan 1 2010 through Jun 30 2010 and a randomly generated temperature for each day. Use `rnorm` to generate the temperatures with the means listed below and a standard deviation of 5, then round to the nearest whole number. Hint: you only have to call `rnorm` once if you first create a vector of means.

First we can create the sequence of dates and find out how many days there are per month. Recall from the lecture on dates:

```
date <- seq(from = as.Date("2010/1/1"),
            to = as.Date("2010/6/30"), by = "day")
month <- months(date)
month <- factor(month, levels = unique(month), ordered = TRUE)
```

Then we can create the temperature values and the data frame.

```
specifiedMeans <- c(40, 42, 51, 55, 58, 62)
meanVector <- rep(specifiedMeans, table(month))
temp <- as.integer(rnorm( n = length(meanVector),
                          mean = meanVector, sd = 5))
```

As you may have discovered, specifying `c(40, 42, 51, 55, 58, 62)` for the mean of `rnorm` will reuse that vector sequentially, so we don't get the right mean for each month. We either need to generate each month separately, which is a valid approach some people used, or first create a vector of means of the right length.

b) Calculate the observed means for each month (based on the data you generated) and compare *to the values specified in part a.*

We can use the months vector we created with `tapply` to calculate the means by month. You should find your sample means close to the specified means, and the difference should be by a similar amount for each month.

```
actualMeans <- tapply(temp, month, mean)
specifiedMean - actualMeans
```

c) Check if you have any duplicate temperatures. On which days does this happen? (Note that this answer will change if you regenerate the data)

```
any(duplicated(temperature$temp))
```

This gives a single TRUE or FALSE if there are any duplicated values or not.

```
temperature[duplicated(temperature$temp), 1]
```

This displays the corresponding dates. We could also use `which` here, though it is unnecessary since recall if we specify a vector of TRUE/FALSE for which rows to display, R will only display those corresponding to TRUE (similar to how some conditional statements work).

Remember that passing the entire data frame to `duplicated` checks for duplicated rows. Our rows are all unique because the dates are unique, so this can't detect duplicated temperatures.

d) Now create a second data frame named `observations` which has 3 columns: the dates Jan 1 2010 through Jul 31 2010, but only every other day, the conditions (sunny, cloudy, or partly cloudy) as a factor (how you assign conditions to days is up to you), and the wind speed. Use `rnorm` to generate the wind speed with a mean of 5 and a standard deviation of 3, but change any negative values to 0.

```
date <- seq(from = as.Date("2010/1/1"),
            to = as.Date("2010/7/31"), by = "2 days")
condition <- factor(sample(c("sunny", "cloudy", "partcloudy"),
                           size = length(date), replace = TRUE))
wind <- rnorm(n = length(date), mean = 5, sd = 3)
wind[wind < 0] <- 0
observations <- data.frame(date, condition, wind)
```

I used `sample` to randomly assign conditions, but assigning them sequentially is another possibility.

e) Combine the 2 data frames into a single data frame `weather`, omitting any rows that don't match.

```
weather = merge(temperature, observations, by = "date")
```

f) Calculate the min and max wind speed for each of the conditions, as well as how many days of each condition there were.

```
tapply(weather$temp, weather$condition, range)
table(weather$condition)
```

`range` is a convenient function when you need the minimum and maximum, and can also me used in plotting to specify `xlim` and `ylim`.