FISH 552
Homework 2 Solutions

Question 2

a) Run the command `head(beaver1)` *to get a sense of what the dataset is, and go to the help file for* `beaver1` *to learn more about this data set.*

```
head(beaver1)
?beaver1
```

b) Use one function to compute the mean temperature for inside and outside the retreat.

```
means <- tapply(beaver1$temp, beaver1$activ, mean)
```

Because we want to get the means according to some grouping, and we already have a column `activ` that describes that grouping, we can use `tapply` to calculate both means at once. I've also saved the return value to use later, which is better than retyping the numerical value.

c) Recreate the following plot:

```
plot (1:nrow(beaver1), beaver1$temp, type = "l",
     xlab = "observation #", ylab = "temperature",
     xaxt = 'n', main = "Beaver 1 body temperature")
xaxispts = c( 1, seq(from = 20, to = 100, by = 20),
     nrow(beaver1))
axis( side = 1, at = xaxispts, labels = xaxispts )
lines(x = c(1, nrow(beaver1)), y = rep(means[1], 2),
     col = "gray", lty = 2 )
lines(x = c(1, nrow(beaver1)), y = rep(means[2], 2),
     col = "gray", lty = 3 )
points( x = which(beaver1$activ == 1 ),
     y = beaver1$temp[beaver1$activ == 1],
     col = "green", pch = 20, cex = 2)
legend( "topleft", legend = c("inside mean temperature",
     "ouside mean temperature"), lty = 2:3, col = "gray",
     bty = 'n')
```

Notice I've referred to the mean values I stored in `means` to plot the lines. The function `abline` is also a nice way to add straight lines to a plot. In order to plot points for periods of activity, I subsetted the data and used the `which` function to get the corresponding x and y values. Remember, it's always better to get the values straight from your data instead of retyping them, which is prone to error.

If we also wanted to add the green point to the legend, we can use `NA` to mix point and line types in a legend:

```
legend( "topleft", legend = c("inside mean temperature",
     "ouside mean temperature", "outside activity"),
     lty = c(2:3, NA), pch = c(NA, NA, 20),
     col = c("gray", "gray", "green"), bty = 'n')
```

d) Note that the observation at 22:20 is missing. Add an observation with a temperature of 37.3 and activity outside the retreat at this time.

```
beaver1Updated <- rbind(beaver1, c(346, 2220, 37.3, 1))
```

e) How much did the mean temperature during periods of activity outside the retreat change?

```
mean(beaver1add$temp[beaver1Updated$activ == 1]) - means[2]
```

Question 2

a) Type the following command to get access to the dataset (we'll cover what a library is later):

```
library(MASS)
head(crabs)
```

b) Use one function to compute the mean of frontal lobe size for blue and orange purple rock crabs. Now compute the standard deviation of frontal lobe size for blue and orange purple rock crabs.

```
tapply(crabs$FL, crabs$sp, mean)
tapply(crabs$FL, crabs$sp, sd)
```

A very common issue was to simply use the mean and standard deviation function multiple times. Remember that `tapply` will apply a function to a vector by each categorical grouping. What if this factor had 10 levels instead of 2?

c) Run this line of code

```
 crabs$sp:crabs$sex
```

Explain what's going on here. Call this new factor, `spsex`*.*

```
spsex <- crabs$sp:crabs$sex
```

This command created a factor corresponding to each unique combination of the levels of the two individual factors. This command is equivalent to specifying an interaction in a model statement (which we'll cover later in linear models), but it can be used outside of that context, which is neat.

d) Change the levels of `spsex` *so that they are more informative. So* `B:F` *might be called* `"Blue Female"` *and so on.*

```
levels(spsex) <- c("Blue female", "Blue male",
                   "Orange female", "Orange male")
```

There are lots of ways to approach it, but the `levels` function is probably the most convenient.

Recall that `levels` lists the named character strings of each level of a factor, so we can change the levels by calling the function and assigning a new character vector of convenient names.

e) Use one function to compute the mean of frontal lobe size for each sex and color combination of purple rock crab.

```
tapply(crabs$FL, spsex, mean)
```

f) Use one function with the vector `spsex` *to compute the total number of each sex and color combination of purple rock crab. Call this vector* `crab.counts`*.*

```
crab.counts <- table(spsex)
```

g) Create an informative plot using the `crabs` *data with appropriate labeling.*

Lots of nice plots here.