FISH 552
Homework 1 Solutions

Question 1

Create vectors of data with the following characteristics, using `rep()` and `seq()` when possible.

a) 100 regularly spaced values from -1 to 1. Call this vector `x`.

```
x <- seq(from = -1, to = 1, length = 100)
```

Many people tried to do this with the `by` argument instead. However, `seq(from = -1, to = 1, by = 0.02)` produces a vector of length 101. You can do this with `by = 2/99`, but the `length` argument is more straightforward.

b) The function y = exp{x / 2}, evaluated for all the values in vector `x` created in (a). Call this vector `y`.

```
y <- exp(x/2)
```

c) How many values in `y` created in (b) are above 1?

```
sum(y > 1)
```

d) 5 entries each of "Small", "Medium", "Large", encoded as a factor. Call this vector `size`.

```
size <- factor(rep(c("small", "medium", "large"), each = 5))
```

You could also do this in several steps, first creating the vector, then making it a factor. Something to watch out for is that you need to save the return value of `factor`, or `size` won't be a factor. Remember, functions in R never change the input, so you need to save the return value of a function if you want to change the object that was input.

e) Take the results of (d) and append 2 entries of "Unknown". Call this vector `observedSize`.

Many people noticed that if we simply use `c` or `append` on size, the result isn't what we want:

```
c( size, rep("unknown", 2))
 [1] "3"       "3"       "3"       "3"       "3"       "2"
"2"       "2"       "2"       "2"       "1"       "1"       "1"
[14] "1"       "1"       "unknown" "unknown"
```

R is converting to the lowest common denominator, character data, but unfortunately when it converts the factor, it uses its internal representation rather than the label. There are several ways of dealing with this, but one way is to recreate the factor after appending the new values.

```
observedSize <- factor(c(as.character(size),
    rep("unknown", 2))))
```

An alternative is to change the levels of `observedSize` after creating it.

```
observedSize <- factor(size,  rep("unknown", 2)))
levels(observedSize) <- c("small", "medium", "large", "unknown")
```

Question 2

For this question we will use counts of salmon and other fish from dams in the Columbia River Basin. The file "`fishPassage.csv`" is available on the Data page of the course website.

a) Read the data into R as an object named `fishPassage`.

```
fishPassage <-
read.csv("http://www.fish.washington.edu/classes/fish507/FISH552/
Data_files/fishPassage.csv", header=TRUE)
```

Note that read.csv will return a data frame, so you don't need to do this again to the results.

b) Calculate the maximum and minimum number of wild steelhead to pass Bonneville Dam (`BON`) and specify in which year each occurred.

Again, there are many ways to approach this. First we can get the min and max, being careful to handle the NAs in the data. Some people did this in multiple steps, which is fine. However, if you have a line of code that returned the row index, you should save that as an object, instead of just retyping the row index. If you just looked at the data to specify the indices, practice more with how to get the indices using logical statements.

```
( minPassage = min(fishPassage$Wild.Steelhead[fishPassage$Dam ==
"BON"], na.rm = TRUE) )
( maxPassage = max(fishPassage$Wild.Steelhead[fishPassage$Dam ==
"BON"], na.rm = TRUE) )
```

Next, we can use the max and min values to get the corresponding year.

```
fishPassage[which(fishPassage$Wild.Steelhead == minPassage),
"Year"]
fishPassage[which(fishPassage$Wild.Steelhead == maxPassage),
"Year"]
```

c) Calculate the total number of fish counted at Bonneville Dam (`BON`) in 2007.

```
sum(fishPassage[fishPassage$Year == 2007 & fishPassage$Dam ==
"BON",3:13])
```

d) Create a new data frame called `fishPassage1995BON` that contains only observations from Bonneville Dam (`BON`) from 1995 onwards.

```
fishPassage1995BON <- fishPassage[fishPassage$Year >= 1995 &
fishPassage$Dam == "BON",]
```

e) Create a matrix with 3 columns from the data `fishPassage1995BON` that contains counts of Coho adults, Coho jacks and the proportion of Coho jacks to adults. Name this matrix `cohoPassage`.

```
cohoPassage <- cbind(fishPassage1995BON$Coho.Adult,
    fishPassage1995BON$Coho.Jack,
    fishPassage1995BON$Coho.Jack / fishPassage1995BON$Coho.Adult)
```

f) Compute the mean of each of the variables in `cohoPassage`. Be careful with missing values (NA).

In this case we didn't have any missing values. I used `colMeans`, but you could also take the mean of each column individually.

```
colMeans(cohoPassage, na.rm = TRUE)
```

This is also a place we could use apply.

```
apply(cohoPassage, 2, mean)
```

g) Round the output in f) to two decimal places. You may need to search the help files.

```
round(colMeans(cohoPassage, na.rm = TRUE), digits = 2)
```

h) Display all the observations associated with the largest Chinook adult count from the `fishPassage` data.

This demonstrates the `which.max` function, though you also could have used the `which` function.

```
fishPassage[which.max(fishPassage$Chinook.Adult),]
```