Stat-Lab 1 Solutions.

Chapter 1: Data/R Basics

1. Nothing to turn in

2. Getting Started/Using Help: Please take a look at p.14-15 of the handout.
Use R to do the following:
(a) 2+5-(3*6)
(b) 3^6
(c) e^2; hint: help(exp)
(d) y <- 3 (or y = 3)
x <- 6*y + 5 (or x = 6*y + 5)
x

```
> 2+5-(3*6)
[1] -11
> 3^6
[1] 729
> exp(2)
[1] 7.389056
> y<-3
> x<-6*y+5
> x
[1] 23
```

3. Help Documentation Files: Please take a look at p.5-15 of the handout.
Use R to do the following:
(a) Find the help documentation for the function quantile. This function takes a list of numbers, a vector, and computes quantiles for the list. What is the description of the probs argument?

```
> help(quantile)
```

The description of the probs argument is: numeric vector of probabilities with values in [0,1]
(b) Find the help documentation for the function mean. This function takes a list of numbers, a vector, and computes their average. What is the example code at the bottom of the help page?

```
> help(mean)
```

The example code at the bottom of the help files is:

```
x <- c(0:10, 50)
xm <- mean(x)
c(xm, mean(x, trim = 0.10))
mean(USArrests, trim = 0.2)
```

(c) Use the help pages to find the name of the function in R that finds the standard deviation of a vector.

```
> help.search("standard deviation")
```

The function is sd(x) where x is the vector.

4. Loading a Library: In R, there are several libraries or packages/groups of programs that are not used enough to have them be stored in R. We can load the library into R by

typing library(library-name) at the command line.
(Sometimes we need to download the library first; more on this later.)
(a) Load the MASS library into R. Open the help documentation for the MASS package (see the Help menu). What is the official name of this MASS package?
> library(MASS)
The official name is “Main Package of Venables and Ripley’s MASS”.
(b) Find the help documentation for the UScereal data in the MASS package. Describe this data set using information from the help pages.
The UScereal data frame has 65 rows and 11 columns. The data come from the 1993 ASA Statistical Graphics Exposition, and are taken from the mandatory F&DA food label. The data have been normalized here to a portion of one American cup. The original data are available at http://lib.stat.cmu.edu/datasets/1993.expo/.
(c) Load the graphics library into R and open its help documentation. This library is full of graphics/visualization functions that we will use in this class. Find a function that creates a Cleveland Dot Plot. Describe its argument x.
> library(graphics)
Could do: > help.search("Cleveland Dot Plot") Or could scan the help pages of the graphics library in the list of functions for something that creates the dot plot.
The function is called dotchart.
> help(dotchart)
The argument x is described as: either a vector or matrix of numeric values (NAs are allowed). If x is a matrix the overall plot consists of juxtaposed dotplots for each row.

5. Reading in Data: Please see p.25 of the handout. We will use read.table to read in a data set from Blackboard. First, download the data set lab1.txt to your Desktop. To use read.table, we either need to know where to locate the data or we can change the directory of R to be the Desktop as well. Type getwd() at the command line to see where the current R directory is; on a PC, you can easily change it by going to the File menu and choosing Change Directory. Use the Browse button to set the Desktop as the new working directory.
> getwd() will be different for everybody; switch the directory if needed
Read the data set into R by typing: lab1data<-read.table("lab1.txt").
Sometimes we need to check that R read it in as a matrix, particularly if it’s one-dimensional. If is.matrix(lab1data) is FALSE, we can change it by typing: lab1data<-as.matrix(lab1data).
NOTE: if the first line of the data set has names for each of the columns, we could read the names in by using: lab1data<-read.table("lab1.txt",header=T)
Type lab1data at the command line to make sure you have the data.
For example:
> lab1data<-read.table("Desktop/315Graphics/Lab1-1-18/lab1.txt")
> lab1data
> is.matrix(lab1data)
[1] FALSE
> lab1data<-as.matrix(lab1data)

6. Data Management: Please see p. 16-25 of the handout

(a) The c( ) function concatenates numbers together into a list. i.e. temp<-c(1,3,6,9) creates a vector named temp that is four elements long and has the elements 1,3,6,9 in that order. Use c( ) to create a vector named temp that contains all the square numbers 1,4,9,... that are _ 100.

```
> temp<-c(1,4,9,16,25,36,49,64,81,100)
```

Could also do:

```
> temp<-seq(1,10)
> temp<-temp^2
```

Use the mean function to find the average of the values in the vector temp.

```
> mean(temp)
[1] 38.5
```

(b) Our data set lab1data is a matrix. Use the function dim to find the dimensions of this matrix (help(dim)).

```
> dim(lab1data)
[1] 9 8
```

(c) In R, we access the rows of a matrix using matrixname[---,] where the —represents the row(s) of interest. if we want more than one row, we use the c( ). Assign the second row of the data to a variable named row2. Use the summary function to find the minimum and maximum of the numbers in row2.

```
> row2<-lab1data[2,]
> row2
V1 V2 V3 V4 V5 V6 V7 V8
9 3 2 3 3 3 9 3
> summary(row2)
Min. 1st Qu. Median Mean 3rd Qu. Max.
2.000 3.000 3.000 4.375 4.500 9.000
```

(d) Columns are accessed using matrixname[,---] where the — represents the column(s) of interest. Assign the third column of the data to a variable named col3. Use the summary function to find the median of the numbers in col3.

```
> col3<-lab1data[,3]
> summary(col3)
Min. 1st Qu. Median Mean 3rd Qu. Max.
1.000 4.000 4.000 4.778 6.000 9.000
```

(e) For one element of the matrix, we provide the row number and the column number, i.e. x<-matrixname[r,c]. Find the element of lab1data in the 5^{th} row, 2^{nd} column.

```
> lab1data[5,2]
[1] 4
```

(f) We can also ask questions about our data/variables. See p. 23-24. The "==" sign indicates a True/False question about the data. e.g. row2==2 returns True for every element in row2 that is equal to 2 and False for elements different than 2. We can do similar things with the symbols <, <=, >, >=.

sum(row2==2) counts how many 2's there are in row2.

which(row2==2) tells you the positions of the elements that are equal to 2.

We can combine these conditions with other variables.

col1[col3==1] lists ONLY the values in the vector col1 if the corresponding values of col3 are equal to 1. For example, the third value of col1 is only listed if the third value of col3 is equal to 1. We can compute things about these subsets as well. mean(col1[col3==1]) computes the average col1 value only for the col1 values for whom the corresponding col3 values are 1.

Create a variable for the first column in lab1data named col1.

```
> col1<-lab1data[,1]
```

Find the mean of the values in col3 whose col1 values are equal to 4.

Just need the last line of code; the rest is there to show the details

```
> col1==4
1 2 3 4 5 6 7 8 9
FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE TRUE
> col3
1 2 3 4 5 6 7 8 9
4 2 4 9 6 1 4 6 7
> col3[col1==4]
9
7
> mean(col3[col1==4])
[1] 7
```

Find the minimum of the values in col1 whose col3 values are less than 6.

Again just need the last line of code

```
> col3<6
1 2 3 4 5 6 7 8 9
TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE FALSE
> col1
1 2 3 4 5 6 7 8 9
3 9 8 10 3 1 6 2 4
> col1[col3<6]
1 2 3 6 7
3 9 8 1 6
> min(col1[col3<6])
[1] 1
```

NOTE: when your data values are text, i.e. letters or words, to use the == sign, you need to put quotes around the text. For example, col1=="A"