Quiz Preparation

Basic data foundations:

Measure of dispersion –
standard deviation, the measure of spread of data about the mean
range – difference between largest and smallest observation in data
interquartile range – the difference between the 25th and 75th percentile, the middle 50% of observations

Law of large numbers – if you run a statistical process a large number of times, it will converge on a stable result

Central tendency - the distribution of sampling means starts to create a bell-shaped/normal distribution and center of that dist gets really close to actual population mean

Independent variable -

Dependent variable -

Vectors:

How to create a vector with a specified list of elements (numbers, etc). Do basic functions on a vector of numbers (ex. sum the numbers).

```
#Create a vector
Data <- c(info, info, info)
```

mean() – returns mean
length() – returns number of values in a vector
sum() – returns sum of data in a vector
sum()/length() – returns average of vectors
max() – returns max value
min() – returns min value
sd() – returns the standard deviation, sd(data, na.rm = TRUE) ignores blank data

range = max – min

```
#Write the R code to test if max height is greater than 60 (output "yes" or "no")
if(maxH > 60) print("yes") else print("no")
```

Data frames:

Understand data frames and be able to write R code that outputs a column, a row, or a specific element in the data frame. Be able to add or remove a column from a data frame. Also, be able to add or remove a row from a data frame. Understand how to access the data frame using "row / column indexing", such as using nrows() to return the position of the last row.

data[row, column] – to return a specific row or column

rownames(data[which.max(data$data),]) – returns the row name of a specific row/column
rownames(data) <- NULL – zeros out row names
nrow() – returns number of rows in a data frame
ncol() – returns number of columns in a data frame
data[-row:-row,-column:-column] – removes a row/column or range from a data frame
scale(data or data$data) – scales vectors
data[order(-data$data),] – orders data in some direction by row
head() – returns first five rows
tail() – returns last 5 rows
colnames(data) <- c(“newname”, “newname”) – renames columns
data <- gsub(“\\.”,””,data$data) – replaces a character with nothing from a column
as.numeric(data$data) – changes column into a number
str() – returns the structure of a data frame
sort(data$data, decreasing = FALSE) – sorts a data frame
replace_na(data, as.list(colMeans(data,na.rm=T))) – replaces NAs with column means

Functions:

Be able to create and use a function to make some calculations (e.g. sum, average)

```
#The function will return the percentage of the elements within the vector that is less than the same.
distStates <- function(myVec, myNum){
  newNum <- myVec[myVec < myNum]
  return(length(newNum) / length(myVec))
}

distStates(dfStates$Jul2011, mean(dfStates$Jul2011))

printVecInfo <- function(X){
  meanX <- mean(X)
  medianX <- median(X)
  minX <- min(X)
  maxX <- max(X)
  sdX <- sd(X)
  quantileX <- quantile(X, probabiliy=c(0.05,0.95))
  skewX <- skewness(X)
  cat("mean:", meanX,
      "median:", medianX,
      "min:", minX,
      "max:", maxX,
      "standard deviation:", sdX,
      "quantiles:", quantileX,
      "skewness:", skewX)
}
```

Understand the quantile function - what is it, why to use it, how to use it

Quantiles divide values into 4 quarters, the median is the middle point (splits group in half)

Summary(data)

Quantile(data, probs=c(0.05,0.95)) or probs=c(0.25,0.50,0.75)– returns quantiles in the quarters you specify. First **shows the chance that the mean would be lower/higher than the returned value**. Second is more precise than what summary() provides.

Understand the sample function - what is it, why to use it, how to use it

Pulls samples from a data set, used to get the distribution, code is

```
sample((data$data), size = 51, replace = TRUE), simplify = TRUE
```

Understand the replicate function - what is it, why to use it, how to use it

Replicate reruns a piece of code as many times as you set, used when running samples to replicate the sample pull to get a large sample of means, code is

```
mean(replicate(100, mean(sample(data$data), size = 51, replace = TRUE)), simplify = TRUE))
```

Understand the histogram function - what is it, why to use it, how to use it

ggplot2

How to use ggplot (and the related plots), including with maps.

```
install.packages("ggplot2")
library(ggplot2)
```

ggplot(data, aes(x=column)) + geom_histogram() – returns a histogram of a column
ggplot(data, aes(y=column)) + geom_boxplot() – returns a boxplot
ggplot(data) + geom_line(aes(x=ccolumn,y-column)) – returns a line
ggplot(data=dataframe, aes(x=column, y=column, color=column)) + geom_line() + stat_smooth() – returns a heat map of the data
ggplot(data) + geom_point(aes(x=column, y=column, size=column, color=column)) – returns a scatter chart

What are the components of the plot functions when using ggplot

The principal components of every plot can be defined as follow:

- **data** is a data frame
- **Aesthetics** is used to indicate x and y variables. It can also be used to control the **color**, the **size** or the **shape** of points, the height of bars, etc…..
- **Geometry** defines the type of graphics (**histogram**, **box plot**, **line plot**, **density plot**, **dot plot**, ….)

How to create a map in R if you were provided with a list of cities and their population

Load the zipcode package to add lat/long info for city based on zipcode

```
data(zipcode)
data$zip <- clean.zipcodes(data$zip)
dataNew <- merge(data, zipcode, by=”zip”)
```

```
> #3. Show the US map, representing the color w/ avg median income of that state
>
> us <- map_data("state")
>
> mapIncome <- ggplot(hm7Simple, aes(map_id=statename))
> mapIncome <- mapIncome + geom_map(map=us, aes(fill=hm7Simple$income))
> mapIncome <- mapIncome + expand_limits(x=us$long, y=us$lat)
> mapIncome <- mapIncome + ggtitle("average median income by state") + theme(plot.title =
element_text(hjust = 0.5))
> ditch_the_axes <- theme(
+   axis.text=element_blank(),
+   axis.line=element_blank(),
+   axis.ticks = element_blank(),
+   panel.border = element_blank(),
+   panel.grid = element_blank(),
+   axis.title = element_blank()
+ )
> mapIncome <- mapIncome + guides(fill=guide_legend(title="Income")) + ditch_the_axes
> mapIncome
```

Linear models:

How to create a basic model and how to measure quality of the model (in our readings)

lm(y ~ x, data=data) – creates a linear model to predict y as a function of x

Summary(lm) – returns the statistical data of the linear model

predict(lmname, data.frame(x = #)) – returns the prediction of y based on x input

lm(y ~ x + x, data=data) – creates a linear model using more than 1 x variable

predict(lmname, data.frame(x = #, x = #)) – returns the prediction of y based on 2 x inputs

Quiz format:

- 1 hour, timed
- 21+ questions, some question are multi part
- You don't need to be in R
- Question examples
  - You'll be given some R code
  - You will be asked questions about the code ie
    - What does this line of code do
    - What will the result be
  - You will be asked to write some R code
  - The R code you write doesn't have to be syntactically correct but close
  - There will be some "concept/topical" questions as noted above