Stat-341 – Term Test 2 – 2016 Spring Term

Part 1 - Multiple Choice

Enter your answers to the multiple choice questions on the provided bubble sheets. Each of the multiple choice question is worth 1 mark – there is no correction for guessing. Be sure your student name and number are completed on the bubble sheets.

1. Which output comes from the following commands?

```
library(plyr)
plotdata   <- data.frame( x=seq(.1,5, .1))
plotdata$y <- laply(plotdata$x, function(x){3+2*x})
head(plotdata)
```

(a)
```
    x   y
1 0.1 3+2*.1
2 0.2 3+2*.2
3 0.3 3+2*.3
4 0.4 3+2*.4
```

(b)
```
    x   y
1 0.1 3.2
2 0.2 3.4
3 0.3 3.6
4 0.4 3.8
```

(c)
```
   y
1 3.2
2 3.4
3 3.6
4 3.8
```

(d)
```
    x   y
1 0.1 0.1
2 0.2 0.2
3 0.3 0.3
4 0.4 0.4
```

(e)
```
   x   y
1 0.1 5.0
2 0.2 6.0
3 0.3 7.0
4 0.4 8.0
```

Solution: (b)
Option A - 14% chose 2016. *R* would not store the unevaluated object.
Option B - 85% chose 2016.

2. Which s output from the following commands?

```
x <- c(5, 3, 1, 4)
order(x)
```

(a) `[1] 1 3 4 5`

(b) `[1] 4 2 1 3`

(c) `[1] 5 4 3 1`

(d) `[1] 3 2 4 1`

(e) `[1] 1 4 2 3`

Solution: (d)
Option A - 22% chose 2016. Order doesn't sort the values.
Option B - 47% chose 2016. Order returns the permutation to sort the data and not the sorted values.
Option D - 28% chose 2016.

3. Which of the following is the proper way to convert the dates of the two term tests into *R*'s internal format.

```
testdates <- c("2016/mar/22", "2016/apr/07")
```

(a) as.Date(testdates, "%Y/%b/%d")

(b) as.Date(testdates, "%Y/%m/%d")

(c) as.Date(testdates, "%y/%b/%d")

(d) as.Date(testdates, "%Y-%b-%d")

(e) as.Date(testdates, "%Y-%m-%d")

Solution: (a)
Option A - 74% chose 2016.
Option B - 16x% chose 2016. The month is not numeric, so %b must be used.
Option C - 2% chose 2016. The year is 4 digits so %Y must be used.

©2016 Carl James Schwarz

Option D - 7% chose 2016. The separator is a slash and not a dash.
Option E - 1% chose 2016. The separator is a slash and not a dash, and the month is not numeric, so %b must be used.

4. Consider the following output:

```
> test
      Name T1 T2
1     Carl 25 22
2     Lois 27 NA
3 Marianne NA 27
4  Matthew 28 NA
5    David 29 30

..... what code goes here to give ....

       Name Test Grade
1      Carl   T1    25
2      Lois   T1    27
4   Matthew   T1    28
5     David   T1    29
6      Carl   T2    22
8  Marianne   T2    27
10    David   T2    30
```

Which of the following code should be inserted above?

(a)
```
melt(test, id.vars="Name",
        measure.vars=c("T1","T2"),
        variable.name ="Grade",
        value.name="Test",
        na.rm=TRUE)
```

(b)
```
melt(test, id.vars="Name",
        measure.vars=c("T1","T2"),
        value.name  ="Grade",
        variable.name="Test")
```

(c)
```
melt(test, id.vars="Test",
        measure.vars=c("Name"),
        value.name  ="Grade",
        variable.name="Test")
```

(d)
```
melt(test, id.vars="Name",
        measure.vars="Grade",
        value.name  ="Test",
        variable.name="Test")
```

©2016 Carl James Schwarz

(e)
```
melt(test, id.vars="Name",
        measure.vars=c("T1","T2"),
        value.name  ="Grade",
        variable.name="Test",
        na.rm=TRUE)
```

Solution: (e)
Option A- 9% chose 2016. Reversed specification for *value.name* and *variable.name*
Option B - 19% chose 2016. This keeps missing values.
Option C - 1% chose 2016. Id variable must be the name.
Option D - 1% chose 2016. The measurement variables is not *Grade*
Option E - 69% chose 2016.

5. The *cereals* data frame contains variables on the number of calories per serving, the amount of fat per serving, and the shelf on which it is displayed. Which of the following code fragments computes the average calories per gram of fat for each shelf?

(a)
```
ddply(cereals, "shelf", function(x){
      ratio <- sum(calories)/sum(fat)
      return(data.frame(ratio=ratio))
})
```

(b)
```
ddply(cereals, "shelf", function(x){
     ratio <- sum(x$calories)/sum(x$fat)
     return(data.frame(ratio=ratio))
})
```

(c)
```
ddply(cereals, "shelf", function(x){
     ratio <- sum(cereals$calories)/sum(cereals$fat)
     return(data.frame(ratio=ratio))
})
```

(d)
```
ddply(cereals, c("calories","fat"), function(shelf){
     ratio <- sum(shelf$calories)/sum(shelf$fat)
     return(data.frame(ratio=ratio))
})
```

(e)
```
ddply(cereals, c("calories","fat"), function(x){
     ratio <- sum(x$calories)/sum(x$fat)
     return(data.frame(ratio=ratio))
})
```

Solution: (b)
Option A- 8% chose 2016. *calories* and *fat* only available in a data frame.
Option B - 67% chose 2016. This keeps missing values.
Option C - 20% chose 2016. The chunk name is *x*.
Option D - 1% chose 2016. The chunks are defined by *shelf* and not by

calories or fat.
Option E - 3% chose 2016. The chunks are defined by *shelf* and not by calories or fat.

6. Consider the following *ggplot()* code:

```
cereals$shelfF <- factor(cereals$shelf)
ggplot(data=cereals, aes(x=fat, y=calories))+
    geom_point(aes(color=shelfF))+
    geom_abline(intercept=90, slope=9)+
    geom_smooth(method="lm", color="red")+
    ggtitle(paste("Cereal data frame with ", nrow(cereals)," cereals"))
```

Which of the following is NOT correct?

(a) The values of *fat* are plotted along the X axis.

(b) The *geom_ abline()* function plots the regression fit between calories and fat.

(c) The *geom_ smooth()* plots the best fit line between fal and calories.

(d) The *geom_ point()* plots the points WITHOUT jittering but colors them by the value of the *shelf*.

(e) The *ggtitle()* creates a title with the number of cereals in the data frame included in the title.

Solution: (b)
Option A - 2% chose in 2016.
Option B - 54% chose in 2016. The *geom_ abline()* plots a line with a specified slope and intercept and not the regression fit.
Option C - 31% chose in 2016.
Option D - 7% chose in 2016.
Option E - 10% chose in 2016.

7. Consider the following vector:

```
> cereal_names
[1] "100%_Bran"          "100%_Natural_Bran"     "All-Bran"      "All-Bran_with_Extra_F
[5] "Almond_Delight"   "Apple_Cinnamon_Cheerios"
```

Which of the following is NOT correct output?

(a)
```
> grepl( "Bran", cereal_names, fixed=TRUE)
 [1]  TRUE  TRUE  TRUE  TRUE FALSE FALSE
```

(b)
```
> cereal_names %in% c("100%_Natural","Almond_Delight")
[1] FALSE TRUE FALSE FALSE  TRUE FALSE
```

(c)
```
> cereal_names[ nchar(cereal_names)>20 ]
[1] "All-Bran_with_Extra_Fiber" "Apple_Cinnamon_Cheerios"
```

(d)
```
> cereal_names[ cereal_names > 'Almond']
[1] "Almond_Delight"          "Apple_Cinnamon_Cheerios"
```

(e)
```
> substr(cereal_names,1,6)
[1] "100%_B" "100%_N" "All-Br" "All-Br" "Almond" "Apple_"
```

Solution: (b)
Option A - 6% chose in 2016.
Option B - 22% chose in 2016. The cereal name is "100%_Natural_Bran" and must be matched exactly (the %in% only has "100%_Natural")
Option C - 2% chose in 2016.
Option D - 58% chose in 2016.
Option E - 13% chose in 2016.

8. Consider the following code fragment:

```
> x
[1] "3"    "4"    "5"    "NA"   NA   "10"
```

Which of the following is NOT correct output?

(a)
```
> sum(is.na(x))
[1]  1
```

(b)
```
> length(is.na(x))
[1] 6
```

(c)
```
> length(x)
[1] 6
```

(d)
```
> sum(x == NA)
[1] 1
```

(e)
```
> x > "10"
[1]  TRUE  TRUE  TRUE  TRUE    NA FALSE
```

Solution: (d)
Option A - 6% chose in 2016.
Option B - 51% chose in 2016. *length()* counts all values, including missing values.
Option C - 5% chose in 2016.
Option D - 13% chose in 2016. All comparisons to *NA* yield missing values.
Option E - 26% chose in 2016.

9. Consider the following code fragment:

```
library(car)
x <- c(1,2,3,4,5)
x
```

Which of the following is NOT correct output?

(a)
```
> newx <- recode(x, " 1:3='small'; 3:5='large'")
> newx
[1] "small" "small" "small" "large" "large"
```

(b)
```
> newx <- recode(x, " 3:5='large'; 1:3='small'; ")
> newx
[1] "small" "small" "large" "large" "large"
```

(c)
```
> newx <- recode(x, "2:3='small'; 4:5='large'")
> newx
[1] "1"     "small" "small" "large" "large"
```

(d)
```
> newx <- recode(x, "2:3='small'; 4:5='large'; else=NA")
> newx
[1] NA      "small" "small" "large" "large"
```

(e)
```
> newx <- recode(x, "3=NA; 2:3='small'; 4:5='large'")
> newx
[1] "1"     "small"    "small"      "large"    "large"
```

Solution: (e)
Option A - 8% chose in 2016.
Option B - 23% chose in 2016. The *recode()* function uses the first clause that meets the criteria to give the new value.
Option C - 2% chose in 2016.
Option D - 7% chose in 2016.
Option E - 60% chose in 2016. The *recode()* function uses the first clause that meets the criteria to give the new value.

10. Consider a data frame of assignment grades for students:

```
> df
  Student assignment grade
1       a          1    13
2       a          2    20
3       b          1    10
4       b          2    11
5       b          3    12
6       c          1    17
7       c          2    NA
8       c          3    18
```

©2016 Carl James Schwarz

Which of the following is correct

(a)
```
df["a",]
  Student assignment grade
1       a          1    13
2       a          2    20
>
```

(b)
```
> df[,2]
[1] 1 2 1 2 3 1 2 3
```

(c)
```
> df[3,]
[1] 13 20 10 11 12 17 NA 18
```

(d)
```
> df['grade' , ]
[1] 13 20 10 11 12 17 NA 18
```

(e)
```
df[df$assignment==1, ]
[1]  TRUE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE
```

Solution: (b)
Option A - 11% chose 2015; 6% choose in 2016.
Option B - 78% chose 2015. 93% choose in 2016.

©2016 Carl James Schwarz

Part II - Long Answer

Stat-341 - 2016 Spring Term - Term Test 2

Name

Student Number:

Put your name and student number on the upper right of each of the following pages as well in case the pages get separated.

Answer the following questions in the space provided. Be sure that your answers are legible.

The marks given to these questions are 5, 8, and 7 respectively.

©2016 Carl James Schwarz

1. **Interpretation** - 5 Marks: What factors are associated with how long animals sleep?

   Different animals spent different amount of times sleeping. For example, humans spend about 8 hours/day sleeping, while cats spend about 15 hours/day.

   Are there physiological variables (such as body mass, brain mass, etc.) that might be related to the amount of sleep?

   Data was collected on 3-5 individual from 62 species. For each individual animal, the body mass (kg) and the total sleep (hrs) was measured along with an indicator of herbivory (h = herbivore, i.e. only eats plants; o = omnivore, i.e. eats both plants and meat). To avoid pseudo-replication, the multiple readings from each species were averaged together. The mean total sleep (hr) was regressed on the mean body mass (kg). An initial plot identified two outliers. Consequently, the logarithm of body mass was used instead.

   Selected parts of the output are found at the end of the exam.

   Write a SHORT paragraph here summarizing the results.

   Solution:
   Your answer should touch on the following points:

   - What is the objective of the experiment or survey?
   - The raw data. How many individuals and species measured?
   - What does the preliminary table indicate? Any missing values?
   - Is there evidence of a relationship? The omnibus test information comes from the *lm()* output.
   - Followup – what is the actual slope?
   - Concerns - problems with the study?

   Here is model solution:

   > A study was conducted to look at the relationship between the average time a species sleeps per day and its average body mass. Data was collected on 3-5 individual animals for 62 species. Each animal was observed and its body mass and total sleep was recorded. Because we are interested in the relationship at the species-level, we averaged the total sleep and body mass over individual within the same species. We were unable to collect data on the mean total sleep for 4 species for various reasons.
   >
   > A preliminary plot of mean total sleep (for each species) vs. the corresponding mean body mass showed two large outliers - corresponding to two species of elephants. A revised plot of the mean total sleep vs. the logarithm of mean body mass generally showed a negative relationship with species with larger body

©2016 Carl James Schwarz

masses tending to have lower mean total sleep. There was evidence of a negative relationship ($p < .0001$) with an estimated slope of -0.84 (SE .16), i.e. for every increase in 1 in the log-mean body mass (i.e. a $2.7\times$ increase in body mass), the mean total sleep declines by .84 hrs.

There are some potential problems in the study. Highly related species likely have highly related sleep patterns. For example, there are two elephant species. This may lead to some "double counting" because you essentially have two data points for "elephants" and so elephants would have twice the influence as species that are more evolutionary distinct. The study also ignored herbivory. It turns out that herbivores tend to be larger than omnivores, so perhaps the relationship with body mass is confounded with herbivory.

You could comment on the using species that are highly related (the data are unlikely to be independent); or the difficulty in measuring total sleep for animals (i.e. how do you tell when a cat is sleeping?); etc.

The following are some additional problems, but way beyond the scope of Stat-341.|

While the relationship holds as the species level, it may not hold for individuals (the ecological fallacy) where the relationship among averages does necessarily hold for individuals `https://en.wikipedia.org/wiki/Ecological_fallacy`.

There is also the related problem of "error-in-variables". The average body-mass for a species is only based on three to five individuals and so has a large uncertainty. This violates one of the assumptions of regression where the X variable is assumed to be measured without uncertainty. The usual consequence of "error-in-variables" is attenuation, i.e. estimates of the slope are pulled towards 0. See `https://en.wikipedia.org/wiki/Errors-in-variables_models`. To reduce the uncertainty in the mean body mass for each species, more individual animals should be measured for each species.

Common problems in solutions from students include:

- Don't just give the table values as "facts" – add some interpretation to the information in the table. For example, some students just listed the mean total sleep and mean total biomass of the three species shown in the output. Why?
- Students read the R p-values of 3.53e-06 as $3.53 \times e^{-6}$. It is actually 3.53×10^{-6}. In any case, never report such small p-values. Modern practise is to simply report these as $p < .0001$ because it is unlikely that all of the assumptions necessary to compute such a small p-value (e.g. normality) are exactly satisfied.

- Students attached the *se* to the p-value, e.g. with a statement such as "...there was evidence that the slope was different from 0 ($p < .0001, SE = .0.16$)". The *se* is attached to the estimated slope (see above).
- Avoid jargon such as
  - "The p-value was . $< .0001$ so we rejected the null hypothesis". Phrases such as "Rejecting the null hypothesis" etc. have NO place in a report.

  These types of sentence provide no useful information to the reader.
- Always cite the p-value. Don't just say "There was evidence ... ($p < .05$)". Modern practice is to give the p-value, unless it is very small. So you should say "There was evidence ... ($p = .002$)" or "There was evidence ... ($p < .0001$)".
- A few student didn't understand that the "Residual Standard Error" reported by R in the *summary()* function output is NOT A STANDARD ERROR. It is misnamed, and is actually the residual standard deviation (RMSE), an estimate of the variation of the data values around the regression line.
- Some student said a larger sample size is needed. Here you need to be careful. There are two levels of sample size – the number of species (62) and the number of animals measured per species (three to five). The number of species is likely adequate (given the problem of potential "double-counting"), and the number of animals/species is somewhat moot as the values for individual animals are averaged first in any case. But see my "error-in-variables" problem above.

©2016 Carl James Schwarz

2. **Creating the animal sleep output** - 8 Marks:
Write *R* code that would generate the output for the animal sleep study that is found at the end of exam.

   - The data is available in a file called *sleep-indiv.csv* and has the variable names in the first row.

   Put your code here and overleaf if necessary. It is not necessary to reproduce the headers on the the output, e.g. it is not necessary to reproduce "Part of the raw data".

   Solution: Here is a sample code.

```
sleep.indiv <- read.csv("sleep.indiv.csv", header=TRUE, as.is=TRUE, strip.white=TRUE)


sink("AnimalSleep/animalsleep-R-data-indiv.txt", split=TRUE)
cat("Part of the raw data\n")
head(sleep.indiv, n=9)
sink()


# Compute the mean Total.Sleep and Body.Weight for each species
sleep <- ddply(sleep.indiv, c("Species","HerbOmni"), summarize,
               mean.TS =mean(Total.Sleep, na.rm=TRUE),
               mean.BM =mean(Body.Mass,   na.rm=TRUE))


sink("AnimalSleep/animalsleep-R-data-means.txt", split=TRUE)
cat("Data summarized up to the species level\n")
head(sleep, n=3)
sink()


# Missing values
sink("AnimalSleep/animalsleep-R-missdata.txt", split=TRUE)
cat("Missing values??\n")
sleep.miss <- summarize(sleep,
                    n.species = length(mean.TS),
                    n.miss.TS = sum(is.na(mean.TS)),
                    n.miss.BM = sum(is.na(mean.BM)))
sleep.miss
sink()


# Initial plot
```

©2016 Carl James Schwarz

```
plot1 <- ggplot(data=sleep, aes(x=mean.BM, y=mean.TS))+
   geom_point(aes(color=HerbOmni, shape=HerbOmni))+
   ggtitle("Mean Total Sleep vs Mean Body Mass")
plot1
ggsave(plot1, file='AnimalSleep/animalsleep-R-plot1.png', h=4, w=6, units="in", dpi=30


# Find outliers with BM > 2000 lbs
outliers <- sleep$mean.BM > 2000
sink("AnimalSleep/animalsleep-R-data-outliers.txt", split=TRUE)
cat("Animals with large body mass\n")
sleep[outliers,]
sink()


sleep$logBM <- log(sleep$mean.BM)
plot2 <- ggplot(data=sleep, aes(x=logBM, y=mean.TS))+
   geom_point(aes(color=HerbOmni, shape=HerbOmni))+
   ggtitle("Total sleep vs. log(Body mass)")
plot2
ggsave(plot2, file='AnimalSleep/animalsleep-R-plot-log.png', h=4, w=6, units="in", dpi


# Fit the regression line
lm.fit <- lm( mean.TS ~ logBM, data=sleep)

sink("AnimalSleep/animalsleep-R-lm-summary.txt", split=TRUE)
cat("Summary of fit from the lm")
summary(lm.fit)
sink()


plot2 <- ggplot(data=sleep, aes(x=logBM, y=mean.TS))+
   geom_point(aes(color=HerbOmni, shape=HerbOmni))+
   geom_abline(intercept=coef(lm.fit)[1], slope=coef(lm.fit)[2])
   ggtitle("Total sleep vs. log(Body mass) with fitted line")
plot2
ggsave(plot2, file='AnimalSleep/animalsleep-R-plot-log-fit.png', h=4, w=6, units="in",
```

The *sink()* calls are to create the text output that was included in the exam and are NOT needed for the student solutions.

Comments about student responses:

- Many students did not get the *HerbOmin* variable in the summarized data frame. Whenever a variable takes a single value that varies with

another variable (e.g. herbivory varies at the *species* level), you can put both into the splitting variables in *ddply()*. Alternatively, if you only split by *species*, then you need some care in getting herbivory into the resulting data frame since you want to return only a single value for that species. So the code will look something like:

```
sleep <- ddply(sleep.inviv, "Species", summarize,
               HerbOmni = HerbOmni[1],
               mean.TS = mean(Total.Sleep, na.rm=TRUE),
               mean.BM= mean(Body.Mass, na.rm=TRUE))
```

- Note that the SUMMARIZED data frame (i.e. one line per species) MUST be used in subsequent plots and analyzes. Many students used the *sleep.indiv* data frame.
- When counting missing values, many students used code similar to

  ```
  n.miss.TS = length( is.na( mean.TS) )
  n.miss.BM = sum( mean.BM == NA)
  n.miss.TS = length( mean.TS, na.rm=TRUE)
  ```

  The first code does not work, because *is.na()* returns a vector which is the SAME length as the original data. The second code does not work because a comparsion to NA ALWAYS results in an NA (try it). The third code does not work because the *length()* does NOT have a *na.rm* argument.
- The *color=HerbOmni* can appear EITHER in the *geom_ point()* or the *ggplot()* layer.
- Many students mixed up the order of the variables in the formulae in *lm()* call. The Y variable (the response) appears to the LEFT of the tilde; the X variables (factors, covariates) appear to the right, i.e. the model is specified as $Y \sim X$.
- Several students hard-coded the slope and intercept in *geom_ abline(intercept=11.4338, slope=-0.8440)*, In general, you should avoid hard coding stuff that is readily available from the data as shown in the solutions.
- Several students used both *geom_ jitter()* and *geom_ point()*. This will give TWO plotted points for each data point – you likely don't want to do that.
- You could also use *geom_ smooth(method='lm', se=FALSE)* to get the fitted line on the plot.

©2016 Carl James Schwarz

3. **Who fibs more?** - 7 Marks:
As taken from `http://www.dailymail.co.uk/news/article-2713862/Men-lie-women-study-reveals.html`

> It's official – men really are more dishonest than women.
>
> While women typically avoid being honest to save someone's feelings, men tend to fib to save them money or win an argument. The average man fibs, or avoids having to tell the truth four times a week. In contrast, women stretch the truth three times a week.

Without worrying about the merits of the survey, write some *R* code to analyze the results of the study.

- Read the file. It is a **.csv* file with the variable names in the first row. It has the following variables:
  - Self-identified gender with *0=female*, *1=male* and *-1=did not respond*
  - Subjects birth year (e.g. 1956).
  - Number of lies told on the day previous to the survey date.
- Convert the gender codes to *m*, *f*, and missing as needed.
- Check your recode using the *xtabs()* function. Hint: what do you need to do to include missing values in the table?
- Compute the subjects age (years) based on this year (2016).
- Create side-by-side box plots of the number of lies told on the previous date for each gender. Include both the raw data and the overlaid box-plot. Notches are NOT needed on the box-plot. Don't forget to jitter the data points.
- Plot the number of lies on the previous day by subject age with separate color and symbol for each gender. Fit a SEPARATE line on the plot for each gender as well.
- Write a function that takes a chunk of the data frame and computes:
  - The proportion who told at least one lie on the previous day. Hint: create a 0/1 variable first based on the number of small lies told yesterday.
  - The *se* of the above proportion computed as $\sqrt{\frac{\widehat{p}(1-\widehat{p})}{n}}$ where $\widehat{p}$ is the estimated proportion and n is the sample size.
  - The approximate 95% confidence interval found as $estimate \pm 2 \times se$.
  - The average number of lies told PER WEEK based on the lies told on the previous day. Notice you need to convert from the average number of lies on a single day to the average per week.

©2016 Carl James Schwarz

- Use the above function to compute the above statistics for each gender.
- Plots the estimated proportions of lying for the genders using a barplot along with the 95% confidence interval as error bars.

Put your code here and overleaf if necessary.

Solution: Sample code:

```
library(car)
library(ggplot2)
library(plyr)

lie <- read.csv('Lie/lie.csv', header=TRUE, as.is=TRUE, strip.white=TRUE)
head(lie)

# recode gender to m/f and check the coding
lie$gender.new <- recode(lie$gender,
                  " 0='f'; 1='m'; else=NA                ")
xtabs(~gender+gender.new, data=lie, exclude=NULL, na.action=na.pass)

# subjects age
lie$age <- 2016- lie$yob

# side by side box plots
ggplot(data=lie, aes(x=gender.new, y=n.lies))+
   ggtitle("Number of lies told in the previous day")+
   geom_point(position=position_jitter(h=0.2, w=0.2))+
   geom_boxplot(alpha=0.2, outlier.size=NA)


# number of lies by age with separate line for each gender
ggplot(data=lie, aes(x=age, y=n.lies, color=gender.new, shape=gender.new))+
  geom_point(position=position_jitter(h=0.2, w=0.2))+
  geom_smooth(method="lm", se=FALSE)


# function to compute statistics
plie <- function(x){
   # propoprtion of lies
   p.lie <- mean( x$n.lies > 0)
   p.lie.se <- sqrt(p.lie*(1-p.lie)/nrow(x)   )
   p.lie.lcl <- p.lie - 2*p.lie.se
   p.lie.ucl <- p.lie + 2*p.lie.se
   a.lie.week <- mean(x$n.lie) * 7  # convert to a per week basis

   data.frame(p.lie, p.lie.se, p.lie.lcl, p.lie.ucl, a.lie.week)
```

©2016 Carl James Schwarz

```
}

# make the little report
report <- ddply(lie, "gender.new", plie)
report


# make a barplot of the estimates
ggplot(data=report, aes(x=gender.new, y=p.lie))+
  geom_bar(stat="identity", alpha=0.5)+
  geom_errorbar(aes(ymin=p.lie.lcl, ymax=p.lie.ucl), width=0.2)+
  xlab("Gender")+ylab("P(lie) with 95% confidence interval")
```

Comments about student responses:

- Many students used code such as

  ```
  df$sex <- recode(df$sex, "1='male'; 2='female'; else=NA")
  xtabs(~sex, data=df, exclude=NULL, na.action=na.pass)
  ```

  So how do you tell that you've done the recode properly? You need to look at a table of the old and new values as shown in the key above.

- Many students computed the age using something like:

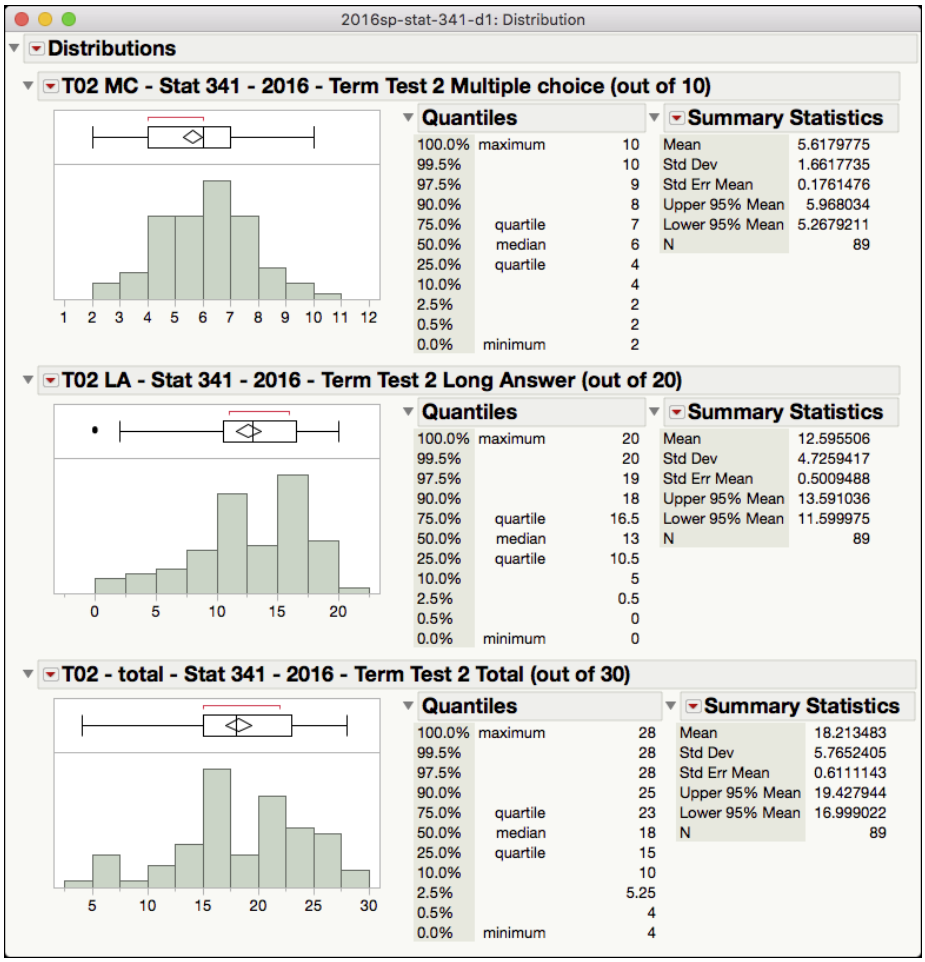
  ```
  df$age <- 2016 - as.Date(df$byear,"%Y%")
  ```

  This won't work because 2016 is not a date object.

- Many students used both *geom_point()* and *geom_jitter()* in the SAME call to *ggplot()*. This will give you TWO points plotted for each data value – this is unlike what you want.

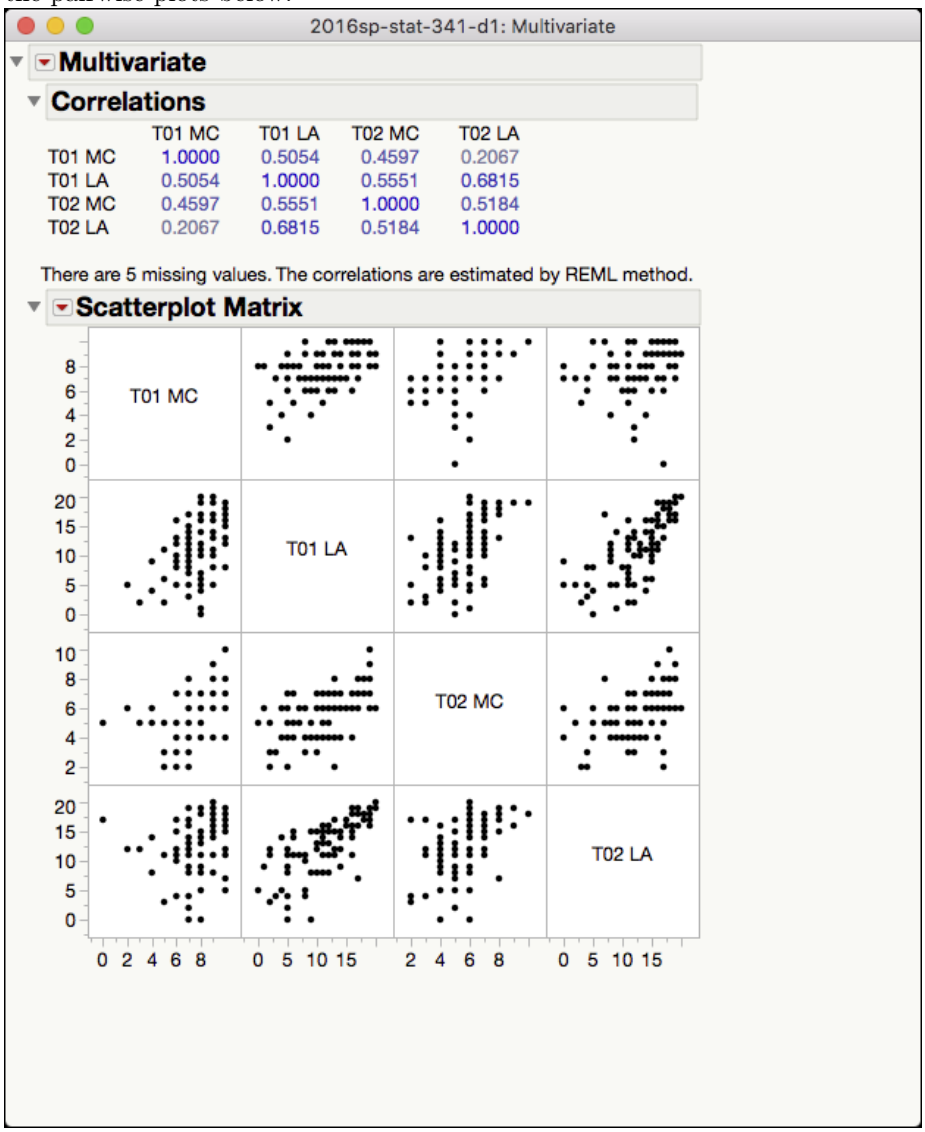
- Many students used *geom_errorbar()* but didn't use *geom_bar()* to get the actual bars. The *geom_errorbar()* only makes the error bars (i.e. the confidence interval in this context), but does NOT draw a bar-chart.

Statistics about the term test:



©2016 Carl James Schwarz

There is some evidence that of relationship between grades on the multiple choice and the long-answer questions on the 4 parts of the term tests as seen in the pairwise plots below.



©2016 Carl James Schwarz

This page intentationally left blank.

©2016 Carl James Schwarz

Output for the animal sleep example - There are 4 pages of output

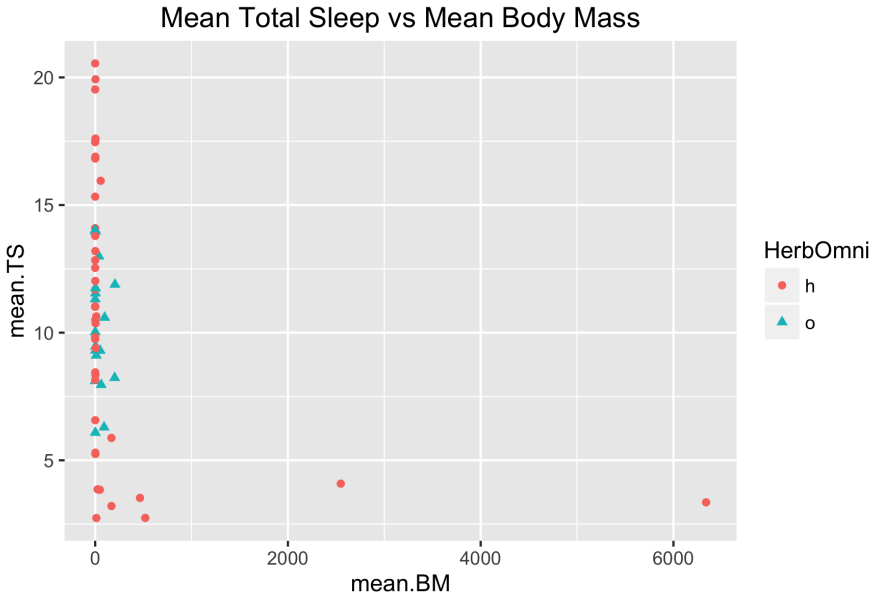
```
Part of the raw data
                   Species HerbOmni Total.Sleep Body.Mass
          African elephant        h       3.466  5799.138
          African elephant        h       3.415  6211.971
          African elephant        h       3.174  7004.843
 African giant pouched rat        o       7.997     1.001
 African giant pouched rat        o       8.029     0.955
 African giant pouched rat        o       8.301     0.922
                Arctic Fox        o      11.116     3.237
                Arctic Fox        o      11.447     3.247
                Arctic Fox        o      12.084     3.323
```

```
Data summarized up to the species level
                   Species HerbOmni   mean.TS      mean.BM
          African elephant        h  3.351667 6338.6506667
 African giant pouched rat        o  8.109000    0.9593333
                Arctic Fox        o 11.549000    3.2690000
```

```
Any Missing values for TS and BM ???
  n.species n.miss.TS n.miss.BM
1        62         4         0
```

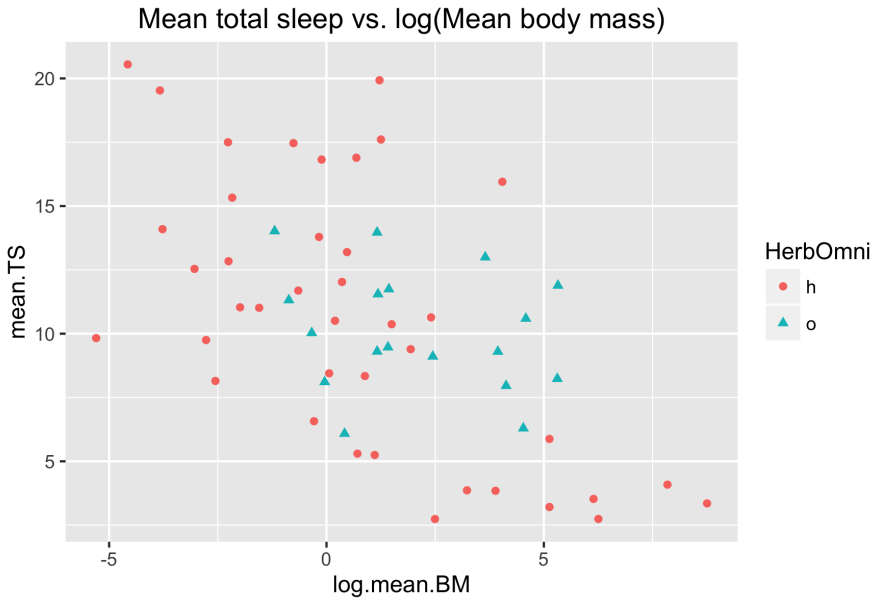
Note: On this and subsequent plots, the herbivory indicator is a different color and symbol.

©2016 Carl James Schwarz




```
Species with large mean body mass
           Species HerbOmni  mean.TS  mean.BM
1 African elephant        h 3.351667 6338.651
5   Asian Elephant        h 4.085333 2548.138
```

©2016 Carl James Schwarz



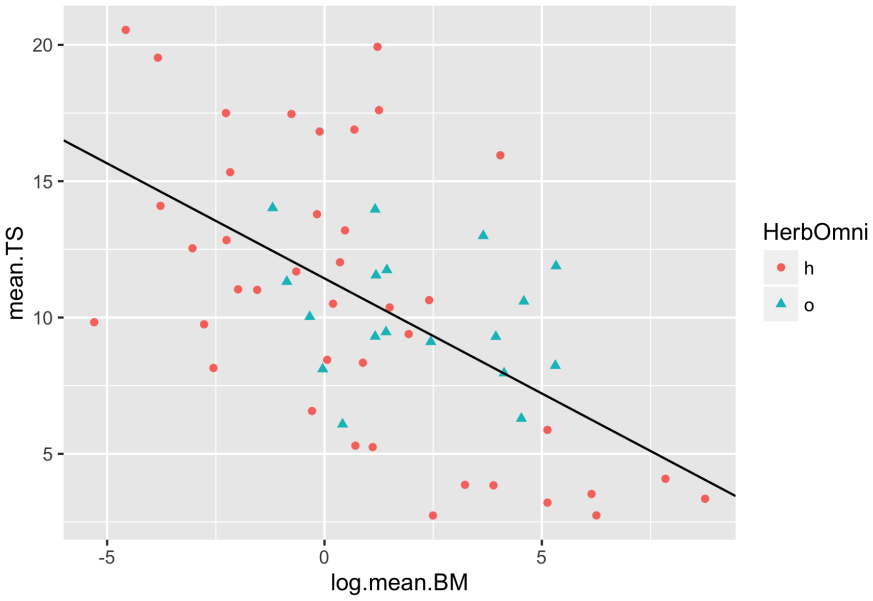

```
Summary of fit from the lm
Call:
lm(formula = mean.TS ~ log.mean.BM, data = sleep)

Residuals:
   Min     1Q Median     3Q    Max
-6.590 -2.629 -0.510  2.137  9.524

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  11.4338     0.5354  21.354  < 2e-16 ***
log.mean.BM  -0.8440     0.1640  -5.147 3.53e-06 ***
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

©2016 Carl James Schwarz

```
Residual standard error: 3.823 on 56 degrees of freedom
  (4 observations deleted due to missingness)
Multiple R-squared:  0.3211,        Adjusted R-squared:  0.309
F-statistic: 26.49 on 1 and 56 DF,  p-value: 3.533e-06
```



©2016 Carl James Schwarz