University of Toronto Scarborough
Department of Computer and Mathematical Sciences
STAC32 (K. Butler), Final Exam
December 12, 2015

Aids allowed:

- My lecture overheads
- Any notes that you have taken in this course
- Your marked assignments and labs, including the midterm exam
- The course R text
- The course SAS text
- Non-programmable, non-communicating calculator

Before you begin, complete the signature sheet, but sign it only when the invigilator collects it. The signature sheet shows that you were present at the exam.

This exam has 42 numbered pages of questions. Please check to see that you have all the pages.

In addition, you should have an additional booklet of output to refer to during the exam. Contact an invigilator if you do not have this.

Answer each question in the space provided (under the question). If you need more space, use the backs of the pages, but be sure to draw the marker's attention to where the rest of the answer may be found.

The maximum marks available for each part of each question are shown next to the question part. In addition, the total marks available for each page are shown at the bottom of the page, and also in the table on the next page.

The University of Toronto's Code of Behaviour on Academic Matters applies to all University of Toronto Scarborough students. The Code prohibits all forms of academic dishonesty including, but not limited to, cheating, plagiarism, and the use of unauthorized aids. Students violating the Code may be subject to penalties up to and including suspension or expulsion from the University.

Last name: _____

First name: _____

Student number: _____

For marker's use only:

Page	Points	Score
1	2	
2	2	
4	2	
5	3	
7	5	
10	10	
12	4	
14	4	
16	9	
17	2	
18	2	
20	7	
21	2	
22	13	
24	5	
25	8	
27	5	
29	9	
30	2	
32	7	
36	1	
37	2	
38	2	
39	2	
40	6	

1. A university employment office wants to compare the time taken by students of different majors to find their first full-time job after graduation. The data are shown in Figure 1. The data came from a spreadsheet and were saved into a file called `job.csv`, in CSV format, in the folder where R is running and also in your "home" folder on SAS Studio. Give code to accomplish the tasks below, in R or SAS as required. Each part can be done with three lines of code or fewer.

   (a) (2 marks) Read the data into a data frame, and display the first 6 (2017: or "few") lines of that data frame (in R).

My answer: I seem to need this first (you can assume that it was already run):

```
R> library(tidyverse)

-- Attaching packages ------------------------------------------ tidyverse 1.2
K ggplot2 2.2.1     K purrr   0.2.4
K tibble  1.3.4     K dplyr   0.7.4
K tidyr   0.7.2     K stringr 1.2.0
K readr   1.1.1     K forcats 0.2.0
-- Conflicts --------------------------------------------- tidyverse_conflic
X dplyr::filter() masks stats::filter()
X dplyr::lag()    masks stats::lag()
```

2015: `read.csv`, since it's a `csv` file, or `read.table` with `sep=","`:

```
R> jobs=read.csv("job.csv",header=T)
R> head(jobs)

    degree days
1 business  136
2 business  162
3 business  135
4 business  180
5 business  148
6 business  127
```

Anything equivalent, like this for the second part:

```
R> jobs[1:6,]

    degree days
1 business  136
2 business  162
3 business  135
4 business  180
5 business  148
6 business  127
```

or even this:

```
R> library(dplyr)
R> slice(jobs,1:6)

# A tibble: 6 x 2
    degree  days
    <fctr> <int>
```

Question 1 continues... This page: _____ of possible 2 points.

```
1 business   136
2 business   162
3 business   135
4 business   180
5 business   148
6 business   127
```

is good. (I didn't talk about `slice` in class; `filter` selects rows based on the content of a variable.)

2017: `read_csv`:

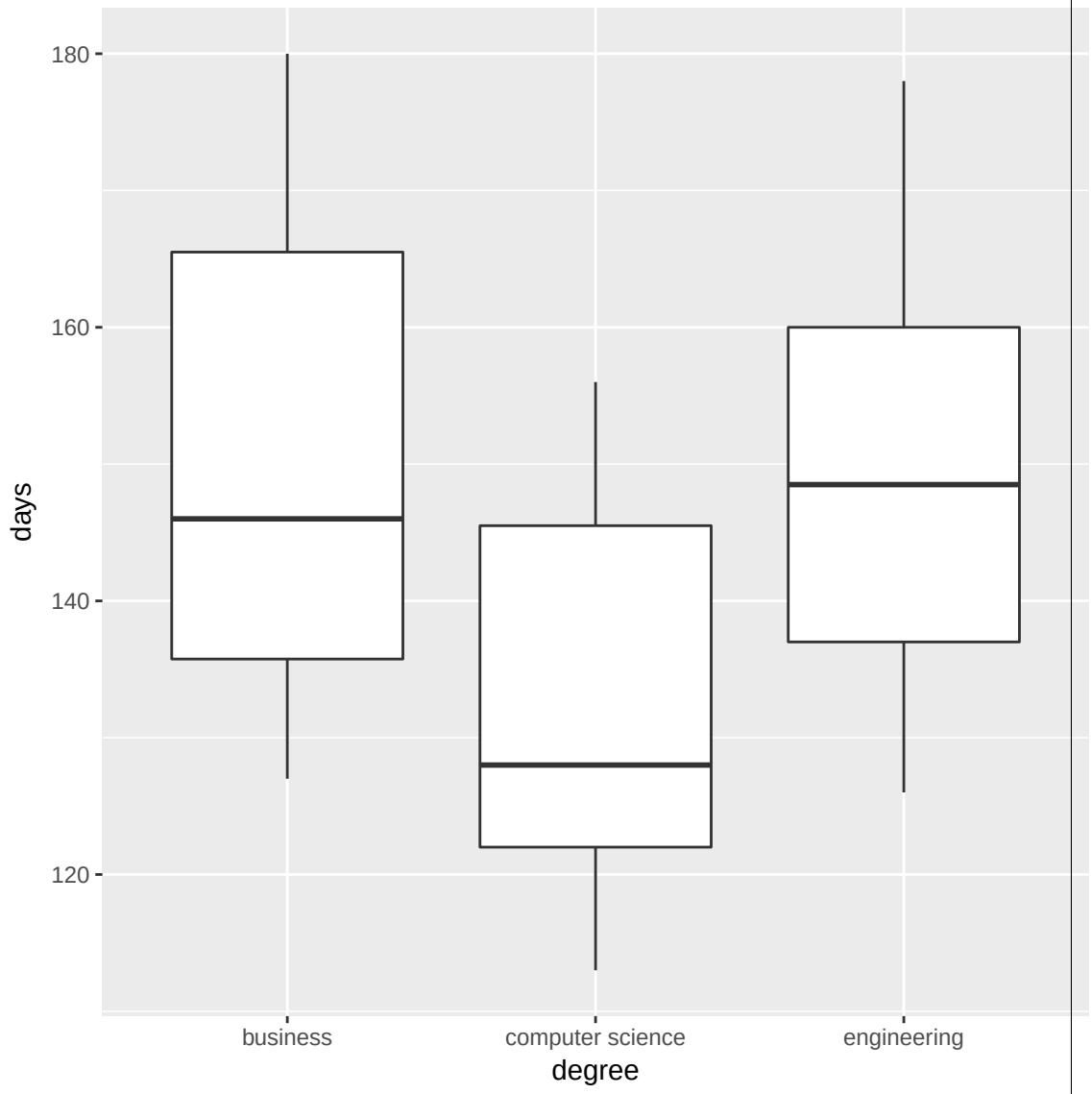
```
R> jobs=read_csv("job.csv")
R> jobs

Parsed with column specification:
cols(
  degree = col_character(),
  days = col_integer()
)
# A tibble: 21 x 2
             degree  days
              <chr> <int>
 1         business   136
 2         business   162
 3         business   135
 4         business   180
 5         business   148
 6         business   127
 7         business   176
 8         business   144
 9 computer science   156
10 computer science   113
# ... with 11 more rows
```

(b) (2 marks) In R, using the data frame that you read in in the previous part, draw side-by-side boxplots of number of days (until a student finds full-time employment) for each type of degree.

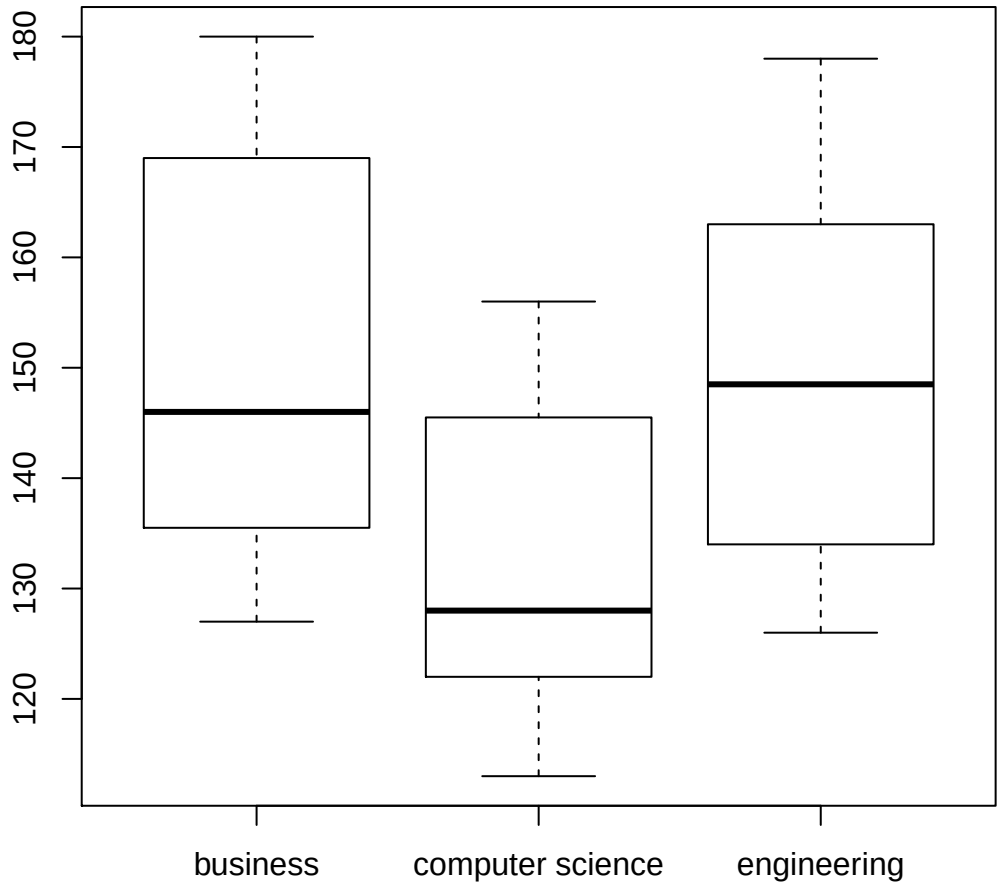
My answer: 2017: the usual:

```
R> ggplot(jobs,aes(x=degree,y=days))+geom_boxplot()
```



2015: The columns are called `degree` and `days`, so use those names:

```
R> attach(jobs)
R> boxplot(days~degree)
```



You need to either `attach` the data frame, or use `data=` in `boxplot`. Saying "I assume that the data frame is attached" is good enough.

(c) (2 marks) In R, obtain the mean number of days until a student finds full-time employment for each type of degree.

My answer: 2017: `group_by` and `summarize`:

```
R> jobs %>% group_by(degree) %>%
R>   summarize(m=mean(days))

# A tibble: 3 x 2
            degree        m
             <chr>    <dbl>
1         business 151.0000
```

```
2 computer science 133.1429
3      engineering 149.6667
```

2015: A job for `aggregate`:

```
R> aggregate(days~degree,jobs,mean)

            degree     days
1         business 151.0000
2 computer science 133.1429
3      engineering 149.6667
```

The model formula first (same one as you used to make the boxplots), then the name of the data frame, then the function you want computed.

If you want to avoid specifying the data frame (eg. because you already attached it), you have to go this way:

```
R> aggregate(days~degree,FUN=mean)

            degree     days
1         business 151.0000
2 computer science 133.1429
3      engineering 149.6667
```

with `FUN` in all caps.

Or even this:

```
R> jobs %>% group_by(degree) %>% summarize(m=mean(days))

# A tibble: 3 x 2
            degree        m
             <chr>    <dbl>
1         business 151.0000
2 computer science 133.1429
3      engineering 149.6667
```

(I already loaded `dplyr` above.)

The story seems to be that computer scientists find a job a little quicker on average.

(d) (3 marks) Read the data into a SAS data set and display that data set. (In SAS.) (2015 note) Do not worry about an "informat" for this question, just read the data as best you can without it.

My answer:

2015: This is not completely obvious, because the data values are separated by commas rather than spaces (and "computer science" has a space *inside* it, but that won't cause any problems).

The online SAS Studio way looks like this:

```
SAS> data jobs;
SAS>   infile '/home/ken/job.csv' firstobs=2 dlm=',';
SAS>   input degree $ days;
SAS>
SAS> proc print;
```

```
Obs     degree     days
  1    business     136
  2    business     162
  3    business     135
  4    business     180
  5    business     148
  6    business     127
  7    business     176
  8    business     144
  9    computer     156
 10    computer     113
 11    computer     124
 12    computer     128
 13    computer     144
 14    computer     147
 15    computer     120
 16    engineer     126
 17    engineer     151
 18    engineer     163
 19    engineer     146
 20    engineer     178
 21    engineer     134
```

The `/folders/myfolders` thing is also good, but you need either that or `/home` and something that looks like a username before the filename. (Use any username you like; I'm not checking.) 2017: this is a routine `proc import`:

```
SAS> proc import
SAS>   datafile='/home/ken/job.csv'
SAS>   out=jobs
SAS>   dbms=csv
SAS>   replace;
SAS>   getnames=yes;
SAS>
SAS> proc print;
```

```
Obs    degree                    days
  1    business                   136
  2    business                   162
  3    business                   135
  4    business                   180
  5    business                   148
  6    business                   127
  7    business                   176
  8    business                   144
  9    computer science           156
 10    computer science           113
 11    computer science           124
 12    computer science           128
 13    computer science           144
 14    computer science           147
 15    computer science           120
```

```
16    engineering                126
17    engineering                151
18    engineering                163
19    engineering                146
20    engineering                178
21    engineering                134
```

No problems here, even with the degree names. (What is happening is that SAS is looking for the longest degree name within the first 20 lines, which *is* the longest degree name altogether.)

(e) (1 mark) (2015) Describe how SAS will read in the degree names without errors, but not as you see them in the data file.

My answer: You'll get only the first 8 characters of the degree names (as I did above). This, however, does not stop you from using those first 8 characters to distinguish the degrees.
The corresponding 2017 discussion is on the end of the previous part.

(f) (2 marks) Obtain the mean number of days required to find full-time employment for a student graduating with each degree type (in SAS).

My answer: Just `proc means` with the right `var` and `class`:

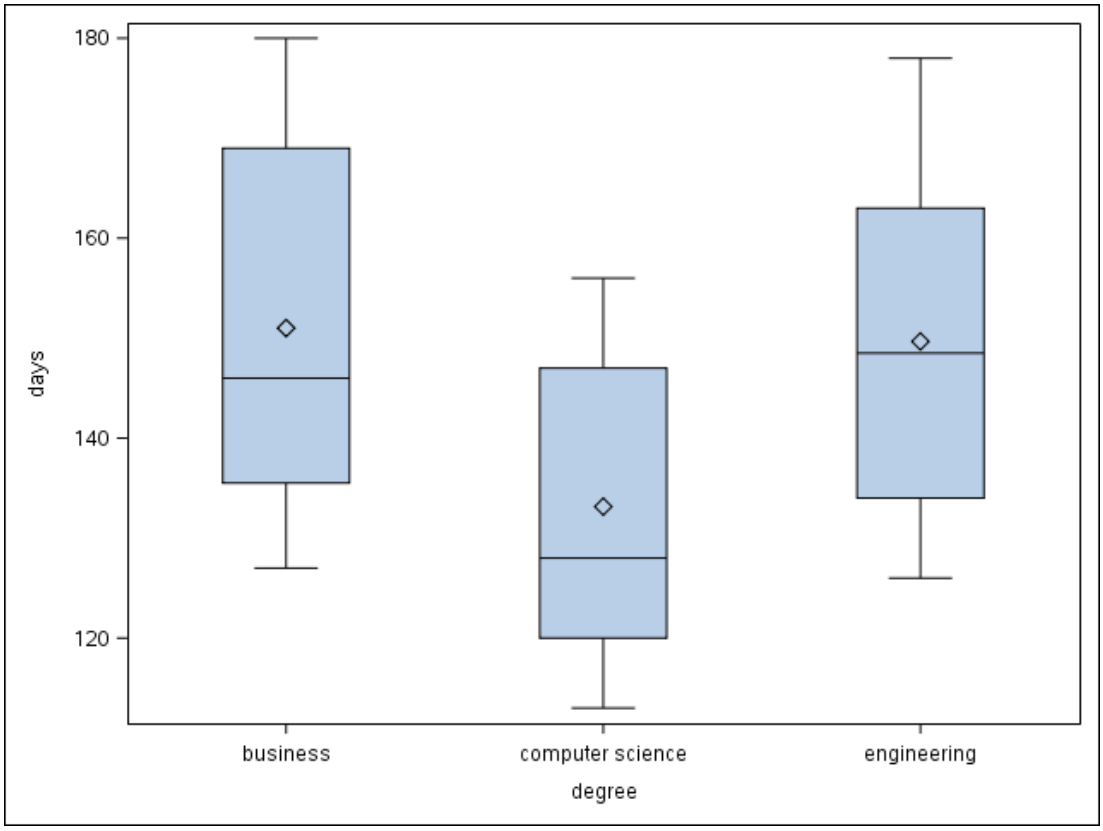
```
SAS> proc means;
SAS>   var days;
SAS>   class degree;

The MEANS Procedure
                                Analysis Variable : days
                      N
degree               Obs     N            Mean         Std Dev         Minimum         Maximum
-------------------------------------------------------------------------------------------------
business               8     8     151.0000000      19.6468827     127.0000000     180.0000000
computer science       7     7     133.1428571      15.9209356     113.0000000     156.0000000
engineering            6     6     149.6666667      18.9806919     126.0000000     178.0000000
-------------------------------------------------------------------------------------------------
```

(g) (2 marks) Obtain side-by-side boxplots of number of days required to find full-time employment for each type of degree, in SAS.

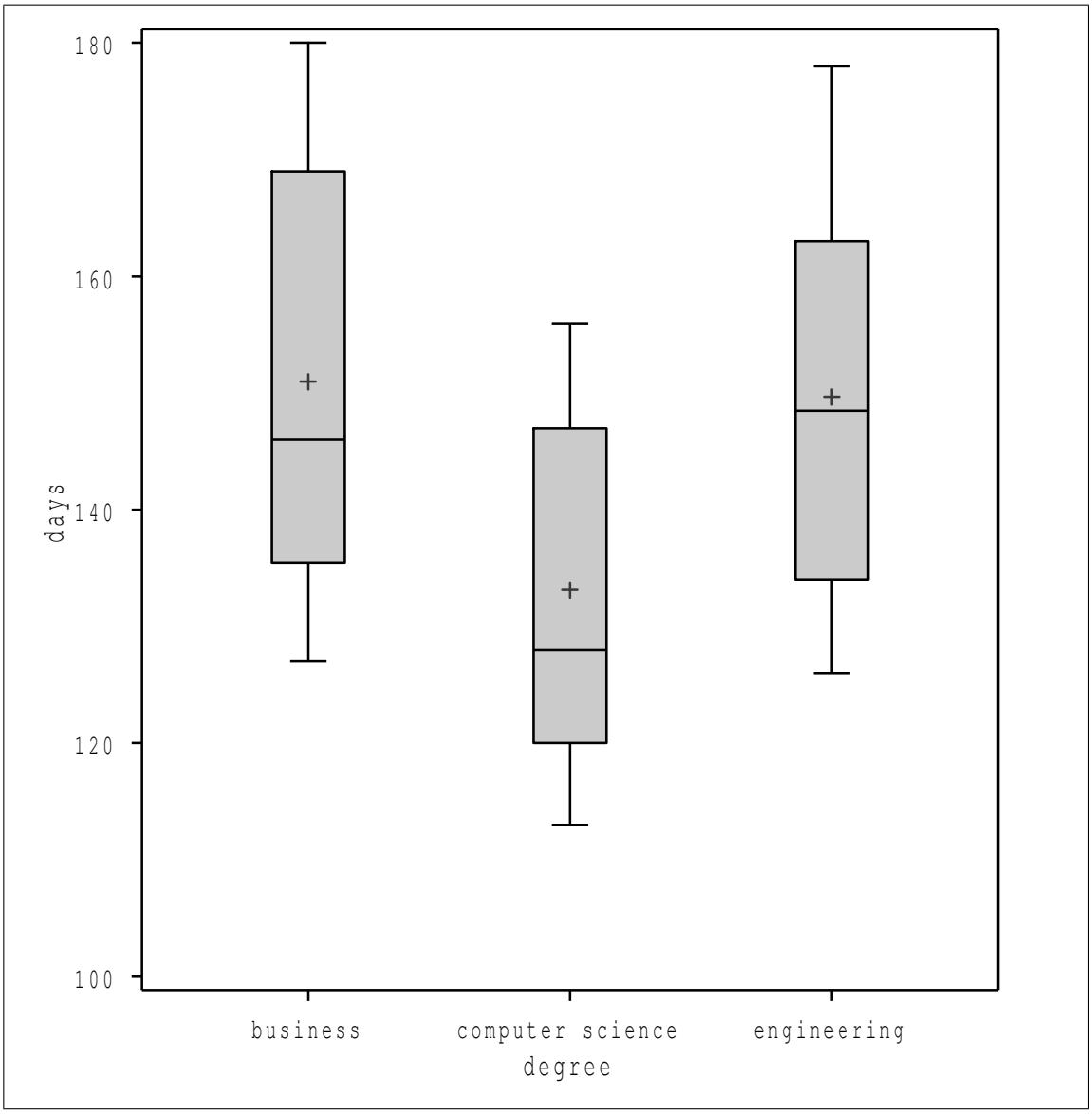
My answer: I did these the other way around from R because the boxplots are (slightly) trickier with SAS and the means are a little easier:

```
SAS> proc sgplot;
SAS>   vbox days / category=degree;
```



The conclusion is the same as from R, even though the quartiles might be slightly different. 2015: Anything else that works, like `proc boxplot`, is also good, if the code is correct, like this:

```
SAS> proc boxplot;
SAS>   plot days*degree / boxstyle=schematic;
```


180
160
140
120
100
days
business
computer science
engineering
degree

2. A farmer decided to start growing blueberries. He bought 8 plants of each of 4 different varieties of highbush blueberries, with the aim of finding out which variety produced the most blueberries. He planted each plant, and measured the yield of blueberries from each one (that is, how many pounds of blueberries each plant produced).

   (a) (2 marks) An analysis of variance is shown in Figure 2. What *null* and *alternative* hypotheses are being tested here?

   My answer: The null is that all four varieties have the same (population) mean yield. The alternative is that the mean yields are not all the same (or that at least one of the mean yields is different from the others, or that at least two of the mean yields are different from each other).

   Your alternative hypothesis must make it clear that the means don't *all* have to be different: thus, "the means are different" is not precise enough.

   (b) (2 marks) What do you conclude from Figure 2, in the context of the data? Explain briefly.

   My answer: The P-value is by no means small, so we cannot reject the null hypothesis, and we therefore conclude that all four varieties of blueberries have the same mean yield.

   (c) (2 marks) Is the analysis of Figure 3 worth doing for these data? If not, explain why not; if it is, explain briefly what you conclude from it.

   My answer: This is Tukey's method. It is designed to reveal which differences in means are significant. But we found in (b) that there are no differences between the means, so it is not worth doing this here.

   If you somehow found significant differences in (b), you should attempt to find out where the significant differences are here, and your failure to find any should be a warning that maybe you should go back and re-examine your answer to (b).

   (d) (2 marks) Look at Figure 4. (2017: this is an old-fashioned boxplot, but the interpretation is the same as a `ggplot` one.) From what you see in this Figure, are you surprised by your conclusions from part (b), or not? Explain briefly.

   My answer: From the ANOVA, we conclude that all four means are equal (not significantly different), and the medians on the boxplots are very close, with a lot of overlap between the groups. So it is completely unsurprising that the ANOVA failed to find any significant differences.

   I don't think you can claim that these medians are different enough to be more than random variation, which is what the ANOVA said.

   (e) (2 marks) Does Figure 4 suggest that there are any problems with the analysis in this question? Explain briefly why or why not.

   My answer: We need to check for two things: that the data in each group are (i) approximately normal with (ii) approximately equal spread. These are both questionable, since the heights of the boxplots (the IQRs, standing in for the SDs) are not at all equal, and there are three outliers (two for Jersey and one for Berkeley), which place doubt on the normality. Having said that, though, the groups with the smaller IQRs are the ones with the outliers, suggesting that the SDs might not be that unequal.

   You need to address two things: "are the data approximately normal within the groups" and "are the spreads approximately equal across the groups". As long as you do that, I don't mind

so much what you say (as long as it's not obviously nonsense).

Let's check the SD thing by finding the means and SDs for each group (the `dplyr` way, just because). First we have to read in the data and massage it a bit, and then we can get to work. (2017: `read.table` was what we used in 2015 instead of `read_delim` and friends to read data from a file. I expect to see you use the right `read_` function this year.)

```
R> blueberry=read.table("blueberry.txt",header=T)
R> blueberry %>% gather(variety,yield,Berkeley:Sierra) %>%
R>   group_by(variety) %>%
R>   summarize(m=mean(yield),s=sd(yield))

# A tibble: 4 x 3
   variety      m         s
     <chr>  <dbl>     <dbl>
1 Berkeley 5.2125 0.1727715
2     Duke 5.1700 0.2753180
3   Jersey 5.1725 0.1472364
4   Sierra 5.2000 0.1903380
```

The means are (as we would expect) extremely close, and the SDs are not that different. Jersey is actually still the smallest, even with the two outliers. So maybe it's not as bad as it looks. (We have only eight observations in each group, so things might look rather jumpy even if the data really were normal with equal SDs within each group.)

You will recognize the `gather` code above as code that *you* had to provide, since Question 8 asked for it.

3. I have an R data frame called `mydata` containing two variables `u` and `v`. I also have a SAS data set called `mydata` containing the same two variables. I want to draw a scatterplot of `v` against `u`, and on that plot, draw the regression line for predicting `v` from `u`.

   (a) (4 marks) Give code for producing this plot in R.

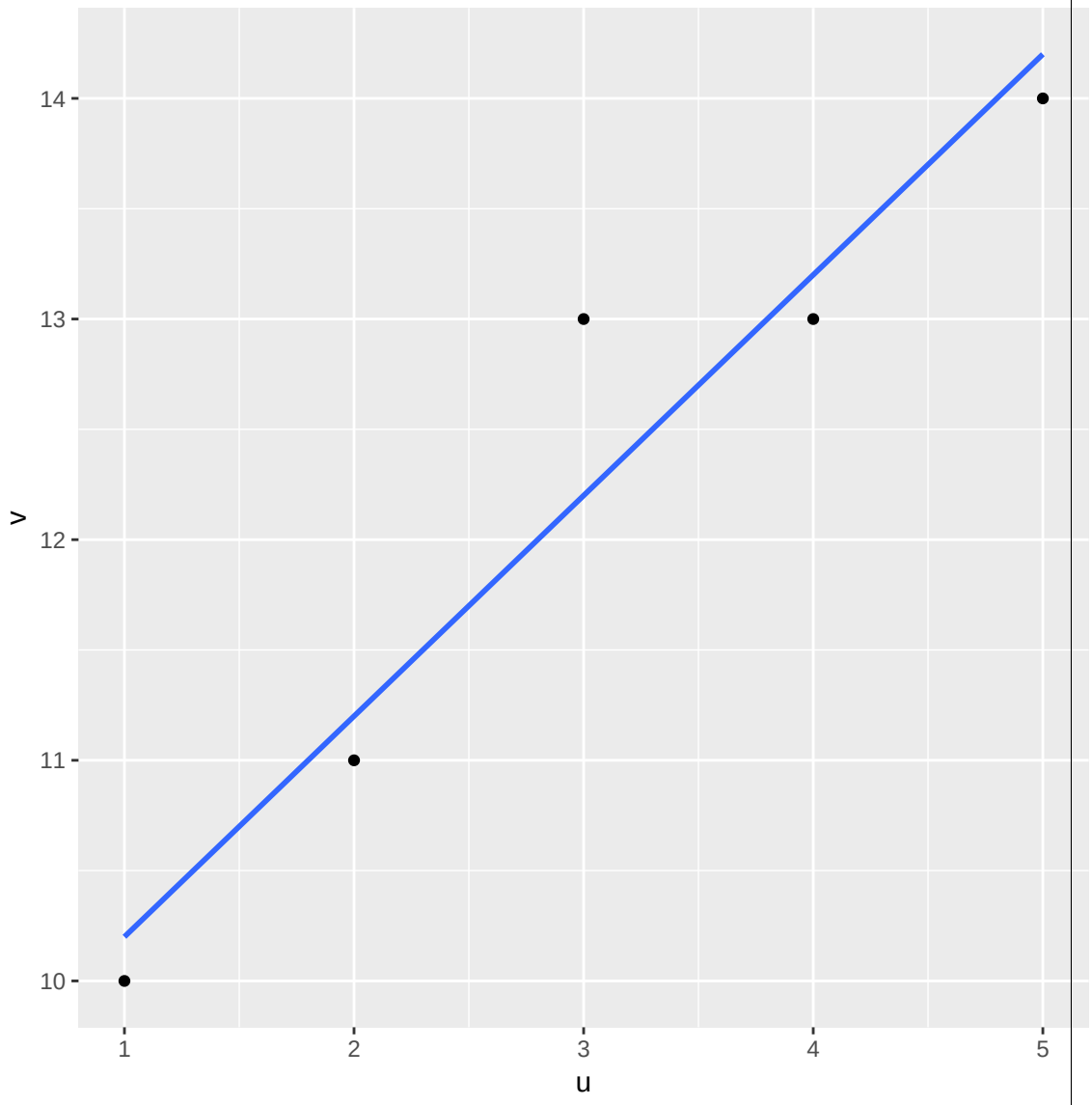
My answer: Let me create a small data set first, so that I can illustrate that my code will work:

```
R> mydata=tibble(u=1:5,v=c(10,11,13,13,14))
R> mydata

# A tibble: 5 x 2
      u     v
  <int> <dbl>
1     1    10
2     2    11
3     3    13
4     4    13
5     5    14
```

2017: this is easier than in 2015, since you can add the regression line to the plot directly:

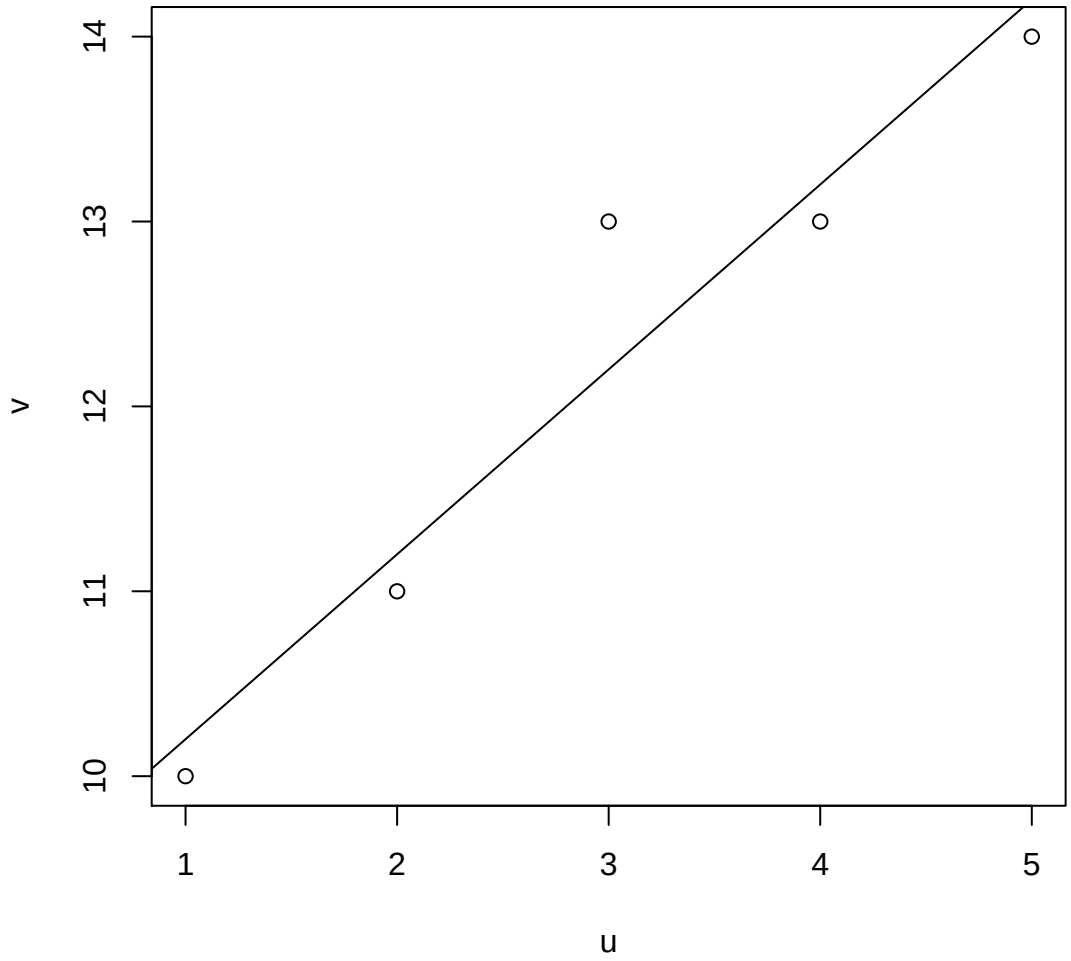
```
R> ggplot(mydata,aes(x=u,y=v))+geom_point()+
R>   geom_smooth(method="lm",se=F)
```



The `se=F` is optional; having it will get just the line, and not the grey "envelope" around it.

2015: And now I create my plot. This bit is what you need. First I have to run my regression, saving the results, then I draw a scatterplot and add the regression line to the plot using `abline`:

```
R>   attach(mydata)
R>   mydata.1=lm(v~u)
R>   plot(v~u)
R>   abline(mydata.1)
R>   detach(mydata)
```



There are other possibilities, but this is the easiest. If what you write works, it is good. You could, for example, obtain the fitted values and plot them against `u` with `lines(spline)`.

(b) (4 marks) Give code for producing this plot in SAS.

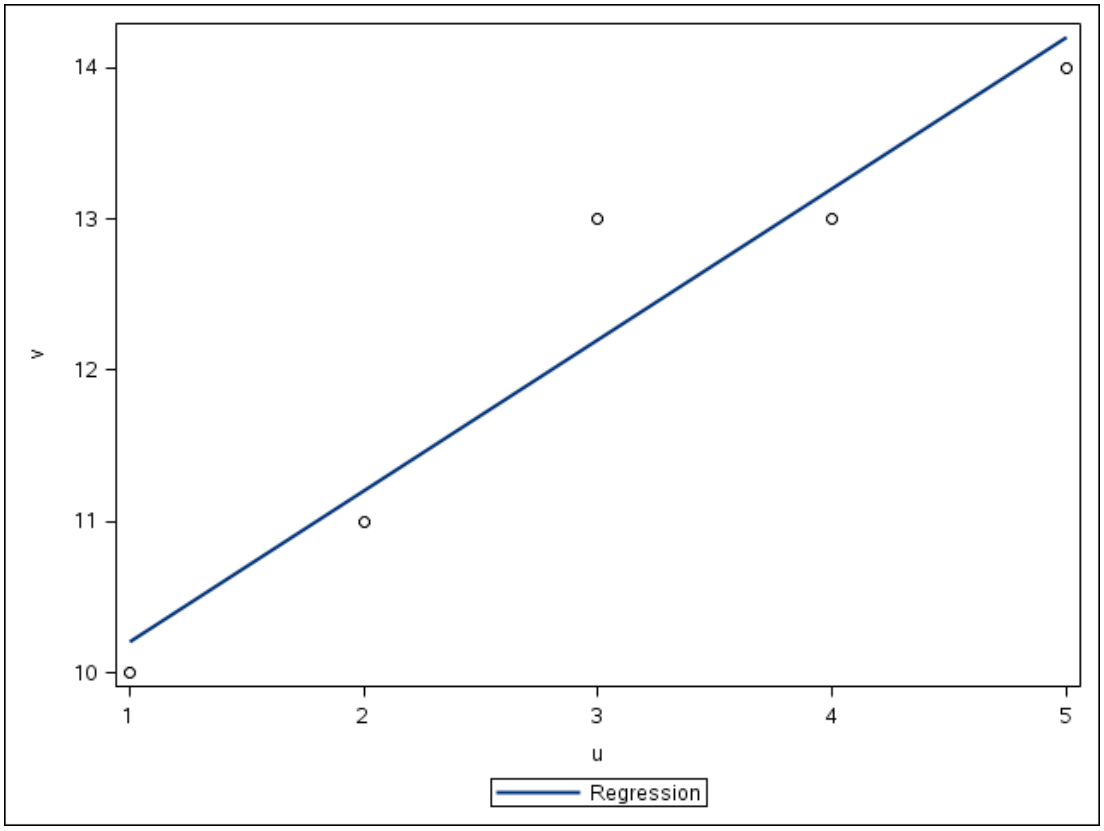
My answer: This is actually easy. The marks are not for navigating the code, but for finding what you need to do.

I create a dataset first: (2017: don't worry about how I did this)

```
SAS> data mydata;
SAS>   input u v @@;
SAS>   cards;
SAS>   1 10 2 11 3 13 4 13 5 14
SAS>   ;
```

And now, the bit that you need. This is `proc sgplot`: not with `scatter` but with `reg`:

```
SAS> proc sgplot;
SAS>   reg x=u y=v;
```



This was in the lecture notes (for electricity usage and peak demand in "data analysis 2"). So the point of this question was for you to remember where you had seen this and to be able to find it. (I don't expect you to be able to remember the code; I didn't.)

A scatterplot *without* regression line gets half marks, if it is correct.

4. Moissanite is a popular abrasive material because of its hardness. It has another important property, elasticity. A mixture containing moissanite was compressed at each of eleven different pressures, and the compressed volume was recorded each time. Some analysis and plots are shown in Figures 5 through 7 in the booklet of code and output.

   (a) (2 marks) In Figure 5, a linear regression model is fitted. What do you conclude from the value `2.83e-10` (2.83×10^{-10}) in that Figure? Explain briefly. (This number appears twice in Figure 5. You may describe your conclusion from either one.)

   My answer: This is the P-value for testing the null hypothesis that the slope is zero. Since the P-value is so small, we conclude that the slope is *not* zero, and therefore that Volume really does depend on Pressure.

   This is also the P-value attached to the F-test, as it will be, given that there is only one explanatory variable. So you can also test the null hypothesis that none of the explanatory variables (there is only Pressure) help to predict Volume. This too is rejected, so that Pressure *does* help to predict Volume.

   Since the same P-value appears in both the t-test for the slope and the F-test for the whole regression, you can give either one of these two answers and you will be good.

   (b) (2 marks) What is R-squared for this regression? What does that tell you? Explain briefly. (Don't look at the adjusted R-squared here.)

   My answer: R-squared is 0.9898, or almost 99%. This is very high, and therefore a straight line describes this relationship very well.

   If you want to go the "percentage variance explained" route, you can do that, but you must get it right: "almost 99% of the variability of volume is explained by the regression with pressure". I don't like this quite as much, because it is possible to say this without really understanding what is going on, but I'll accept it. (The meaning of this is that almost the entire reason that volume varies is that it depends on pressure. Apart from that, there's almost no variability left.)

   (c) (3 marks) What do you learn from Figure 6? (2017: this is a plot of residuals vs. fitted values with a smooth trend.) Is it therefore worth trying the analysis shown in Figure 7? Explain briefly.

   My answer: Figure 6 is a residual plot from the linear regression in Figure 5. This residual plot shows a clear curved pattern, and therefore the relationship between Pressure and Volume is actually curved rather than a straight line. (The smooth curve actually accentuates the curve of the points, but the pattern is pretty clear anyway. I am trying to make it easier for you.)

   The analysis in Figure 7 predicts Volume from Pressure and Pressure-squared, a curved relationship. We have just said that the relationship is actually curved, so this seems like a sensible thing to try. (There might be some physics guiding us towards the correct form of relationship, but in the absence of this knowledge, a quadratic curve has got to be worth trying.)

   (d) (2 marks) Is the second regression, in Figure 7, a statistically significant improvement over the first one? How can you tell? Explain briefly. (You don't need any more output than is already there.)

   My answer: We have added a squared term, so we test this for significance. The P-value is 0.00086, small, so we conclude that the squared term cannot be taken out, or that it is worth adding (or keeping). The quadratic (parabola) model is a significant improvement over the linear one.

   I put the words "statistically significant" in the question to guide you towards doing a test and

getting a P-value. Using the common-English meaning of "significant" will not get you to a clear answer, and nor will comparing the R-squared values of the two regressions. (R-squared doesn't increase much, because it has no room to increase much.)

I could have done an `anova` comparing models `m.1` and `m.2`, but that would have given you the same P-value as the t-test for the slope of the squared term, and therefore the same conclusion. So I didn't need to give you that.

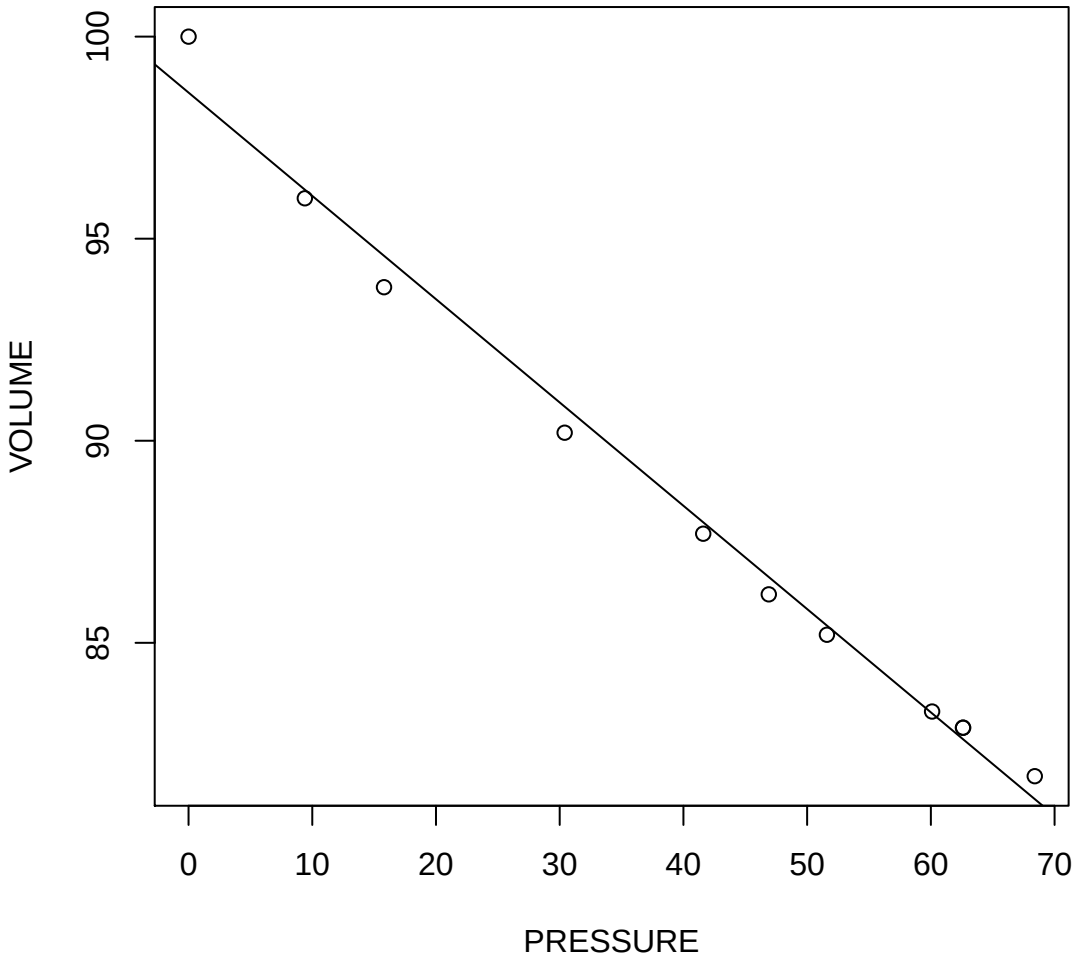
(e) (2 marks) Are your answers to parts (b) and (d) consistent with each other, or inconsistent with each other? Explain briefly.

My answer: Part (b) is saying that a straight line fits very well, and part (d) is saying that a curve fits better. These seem to be inconsistent, but what is actually happening is that even though there is not much room for improvement over a straight line, there is still enough for the parabola curve to be significantly better. The parabola curve actually fits within a hair's breadth of perfectly.

There are lots of ways of getting the marks here. For example, you could talk about how R-squared is a too simple-minded way of assessing fit and we need to look at the residuals before we say anything. I marked this part according to what you thought in the previous part; for example, if you compared R-squareds, which won't have gotten you any marks above, and said there wasn't much difference, then you'd come out with a conclusion here that both models fit very well and therefore the two parts were very much consistent. In that case, this would be fine. As ever, not the answer, but the reasoning behind the answer.

A scatter plot with the regression line on it shows rather nicely what's going on:

```
R> moissanite=read.table("moissanite.txt",header=T)
R> m.1=lm(VOLUME~PRESSURE,data=moissanite)
R> attach(moissanite)
R> plot(VOLUME~PRESSURE)
R> abline(m.1)
R> detach(moissanite)
```



The points are very close to the line (which is why R-squared for the line is so high). But look at how the points deviate from the line: the first one is above, the next *six* are all below, and the last three are above again. So using a curve, even a very slightly curved curve, stands to describe the relationship better. The coefficient of pressure squared in Figure 7 is 0.0013, which looks very close to zero (and indicates that the curvature of the curve is very small), but it is so precisely estimated that it *is* significantly different from zero. Using the usual 2-times-SE for a 95% confidence interval says that we estimate this coefficient to be between about 0.0008 and 0.0018: we are almost certain that it is not zero, but we are also almost certain that it is not very big.

Yes, I know I said eleven points in the question, and there only seem to be ten points on the plot. The regression output is consistent with there being 11 points (look at the degrees of freedom), so my best guess is that two of the points are overwriting each other on the plot.

(f) (2 marks) What do you hope that the residual plot from the model of Figure 7 looks like? Why is that a good thing to hope for? Explain briefly.

My answer: I would hope that it looks like a random mess of points. If it does, that would make me confident that I had found a good model to describe the relationship between Volume and Pressure.

This is about the only kind of positive answer that a residual plot can give you. Seeing a pattern would tell you only that there was some kind of problem; for example, seeing a curve again would tell me only that I hadn't found the right kind of curve to model, but it wouldn't tell me what kind of curve to try next.

I actually *did* draw the residual plot for this second regression, and it still looks a bit curved (and there is also a bit of evidence of fanning out). Because the parabola curve fits *so* well, though, I'm not too worried. My model is giving me excellent predictions (at least within the range of the data). If the quality of fit had been worse, I might have continued to search for a good model, but with things as they are, I'm happy to stop.

5. An environmental researcher is studying carbon monoxide concentrations in air. A carbon monoxide level (in milligrams per cubic metre) is supposed to have a mean of 10 or less; if the mean is more than 10, it is considered to be too high. Carbon monoxide measurements typically have a standard deviation of 1.2.

   The standard way of testing for environmental carbon monoxide pollution is to collect 18 air samples and measure the mean carbon monoxide concentration in all of them. A hypothesis test is carried out, with a null mean of 10, and if the null hypothesis is rejected (in favour of the appropriate one-sided alternative), the location where the samples were taken is declared to be polluted with carbon monoxide.

   The researcher is interested in the "sensitivity" of this procedure: that is, how frequently the location is declared to be polluted *when it actually is polluted*.

   (a) (2 marks) Explain briefly how the P-value of the test *does not* assess the researcher's interest.

   My answer: The P-value measures how likely we are to see the observed result *if the null hypothesis is true*. But in this case, the null hypothesis is *not* true, because the location actually *is* polluted. (That means, looking ahead, that we are interested in the *power* of the test instead.)

   You must mention that the researcher's interest is in the type II error or power. It is true that the P-value usually leads to making a decision about the null, but what is needed here is to recognize that the P-value is the probability of rejecting the null *when it is actually true*, and this is not what the researcher wants: the researcher is playing what-if and asking "if the null is actually false, how likely am I to correctly reject it?". We are thinking of both the test procedure and the power as talking about the *properties* of the test, not about the result that you get from one data set.

   (b) (3 marks) Explain precisely what the output in Figure 8 tells us. You need to talk about carbon monoxide levels in your answer. (I am looking for what you learn *overall* from this output, so I do not need to know what each line of code is doing.)

   My answer: This is calculating the power of the above test when the true mean is 10.50, using a sample size of 18. That is, when the mean carbon monoxide level at the location being tested is really 10.50, the probability of correctly declaring it to be polluted with a sample size of 18 is 0.521.

   Saying "the power is 0.521" is not a complete answer, because you have not shown that you know what the power represents.

   Also, I don't need to know what each individual line says. I want to see that you know precisely what the *whole thing* tells you: what the power is, and *under what circumstances*.

   (c) (2 marks) Figure 9 shows a second analysis. What do you learn from this one?

   My answer: When the true mean is 10.50, it requires a sample size of 38 to obtain a power that is (at least) 0.80. That is, when the true carbon monoxide level at a location is 10.50, it requires a sample of size 38 to have a 80% chance of correctly declaring it polluted.

   I'm willing to accept "it takes a sample size of 38 to obtain a power of 0.80", as long as you've shown in the previous part that you know what the practical importance of this is.

   I aimed in my code for a power of 0.80, but the output tells me that I actually *got* a power of 0.810. (Saying this, instead of 0.80, is good.) This is because the sample size must be an integer; a sample size of 37 would give power *less* than 0.80, so SAS returns the first integer that gives you power *at least* what you wanted. R gives you a decimal sample size:

```
R> power.t.test(delta=0.50,sd=1.2,power=0.8,type="one.sample",
R>   alternative="one.sided")
```

Question 5 continues... This page: _____ of possible 7 points.

```
     One-sample t test power calculation

              n = 37.00253
          delta = 0.5
             sd = 1.2
      sig.level = 0.05
          power = 0.8
    alternative = one.sided
```

You see here that a sample size of 37 *narrowly* fails to give you the power you want, so you round up to get 38.

(d) (2 marks) Explain briefly how Figures 8 and 9 give results that are consistent with each other.

My answer: Figure 9 shows both a larger sample size and a larger power than Figure 8. Other things being equal, if you want to increase the power, you have to increase the sample size. Or, if you increase the sample size without changing anything else, you'll increase the power. Anything roughly like that. It's the usual thing about larger sample sizes being more informative, with "informative" being tailored to the situation.

6. 2017: this is a randomization test, which we didn't do. Your can skip this. How does the brain respond to sounds? A researcher hypothesizes that the brain responds differently to "pure tones" (generated by a computer) than to recognizable sounds. 37 macaque monkeys were anesthetized, and pure tones and monkey calls were fed directly to their brains using electrodes. Response to the stimulus was measured by the firing rate (electrical spikes per second) of neurons in various areas of the brain. Note that each monkey received both kinds of sounds, in randomized order.

   Some of the data are shown in Figure 10. Some analysis is shown in Figures 11 through 14.

   (a) (2 marks) What, in words, is the number `obs` that I calculate in Figure 11?

   My answer: Observed mean difference in firing rate between tones and monkey calls for our data. (The name `obs` is rather a clue.)

   (b) (3 marks) In Figure 12, a function is defined. In that function, `abs(x)` takes the absolute value of `x`, that is, its negative sign (if it has one) is removed. The last line of code shows some examples of running that function. What, in words, is that function doing, and how is it doing it?

   My answer: It is affixing a random sign to the values in `x`. It does this by selecting as many random `+1` and `-1` as there are values in `x`, taking the original values *without* their sign (most of the original data values in `d` were negative), and multiplying to put a random sign on each value. It then calculates the mean of the data-with-random-sign and returns it.

   The "sampling from -1 and $+1$" idea is the same thing we did in class for a randomized matched pairs test, which is what this is.

   I am after the "big picture" here, with an eye on "why is this being done" as well as "what precisely is being done".

   The purpose of the last line is to give you a clue about what is going on: because the function has random numbers in it, it returns a different value, sometimes positive, sometimes negative, each time you run it. (This is actually five values from the randomization distribution.)

   (c) (3 marks) What is the code in Figure 13 doing? (Three things, one for each of the three lines.) Include in your answer for the last line a brief explanation of *why* we want to do this.

   My answer: The first line is doing the randomization 1000 times (running the function above and getting a randomized mean difference), gathering the results together into a vector. The second is drawing a histogram of the results (that's almost a free point), and the third is adding to the histogram a vertical line showing where the observed mean difference comes on the randomization distribution.

   For the last part, "a vertical line at `obs`" is not a complete answer, because it doesn't show me that you know *why* we might want to draw a vertical line there. The purpose of doing that is to see how our data compare to the randomization distribution.

   (d) (2 marks) What does the histogram in Figure 13 tell you about the P-value of your test? Explain briefly.

   My answer: It tells us that the P-value is going to be very small, because the observed value is way out in the tail of the randomization distribution.

   (e) (3 marks) Obtain a P-value from the information in Figure 14. What do you conclude from it, in the context of the original problem?

My answer: P-value is $1/1000 \times 2 = 0.002$ (the researcher is looking for *any* difference, so the test is two-sided). This is less than 0.05 (or 0.01 or whatever), so we can reject the null hypothesis that there is no difference in mean firing rate for the two kinds of sounds, in favour of the alternative that there is a difference. That is, we conclude that the monkeys respond differently to the two types of sounds.

The randomization distribution looks *very* normal in shape, and so you might suspect that the paired t-test will give a very similar result. It does. I got a P-value of 0.0008. (I ought to have run the randomization more times, in order to estimate the randomization P-value more accurately; on my randomization, I couldn't get a P-value between 0.002 and zero.)

7. What does it cost to transport something by truck, and what does that depend on? In 1980, several states had rules controlling trucking within that state, rules that were later removed. Florida was one of the first states to institute a "deregulation" policy. We have data on 134 shipments made by a particular Florida carrier. The response variable of interest is the price charged per ton-mile (`pricptm` in the data set). The logarithm of this variable was previously found to have a more nearly linear relationship with the other variables. These are:

   - `mileage`: distance that the shipment was shipped.
   - `shipment`: weight of the shipment in tons.
   - `pctload`: how full the truck was, in percent of maximum loading.
   - `origin`: `MIA` (Miami) or `JAX` (Jacksonville).
   - `dereg`: `YES` (rules have been removed, deregulation applies) or `NO` (rules are still in effect, deregulation does not apply).

   (a) (3 marks) I obtained the original data as a SAS permanent data set called `trucking`, which I stored in my home folder on SAS Studio. I am going to do a regression predicting log-price per ton-mile from the other variables, which need to be numbers. In Figure 15, I create a new data set. Explain what each of the four lines of code in the `data` step (not including the `data truck2` line) are doing.

   My answer: First, I bring in all the data from the permanent data set. Second, I create a new variable that is log-price per ton-mile, since I need that as the response variable. Third and fourth, `origin` and `dereg` are categorical variables, not numerical. The code here creates 0–1 variables: specifically, `origin01` is 1 if the origin is Jacksonville and 0 otherwise (Miami), and `dereg01` is 1 if the regulations have been removed (deregulation is in effect) and 0 otherwise (the regulations are still in place, or deregulation is not in effect).

   I know this last thing, because the things inside the brackets are logical conditions that are true or false for each observation, and in SAS "true" is 1 and "false" is 0.

   2017: we may or may not get to SAS "permanent data sets": the idea is that instead of reading the data from the file using `proc import` every time, we read the data in *once* and then save the *data set* in a way that we can re-use it later, the same way that R saves a data frame from one session to the next.

   2017 additional: this year, we've looked at `proc glm` which would be able to handle `origin` and `dereg` as categorical variables, rather than turning them into numerical ones as I had to do here (in order to use `proc reg`).

   (b) (2 marks) Figure 16 contains a regression. In that regression, the slope coefficient for `shipment` is 0.2245. How do you interpret that number, in the context of the data?

   My answer: The variable `shipment` is the weight of the shipment in tons, so that the slope says that if you increase the weight of the shipment by 1 ton *without changing anything else*, the log-cost per ton-mile of shipping it increases by 0.2245.

   You need to include the "without changing anything else" or words like "all else equal" (or "allowing for the other variables") because otherwise the other variables could change, and that messes up the interpretation.

   What I found interesting is that I would expect a heavier shipment to cost more to ship in an absolute sense (more actual $), but our response variable is per ton-mile, so it's standardized by size of shipment and how far it's going. The positive slope is saying *when you allow for that*, it's still more expensive per ton-mile to ship something heavier. You could argue that part of the cost is loading the item onto the truck and unloading it at the other end, and a heavier item would cost more to load and unload, but once the item is on the truck, the only additional

cost to taking it further is the driver's pay and the fuel. I suppose you'd argue that an item that is twice as heavy is *more* than twice as awkward to get on the truck in the first place.

On the other hand, it's *less* expensive per ton-mile to ship something a longer distance, because the slope of `mileage` is negative. (This is like GO Transit fares being cheaper *per kilometre* the further you go, even though the actual fare is higher if you go further. GO's reasoning is different, though, because they have a high "base" fare to discourage you from riding a short distance, a journey that you could probably make on local transit.)

(c) (2 marks) Is it more expensive or less expensive to ship an item when deregulation is in effect (as compared to when deregulation is not in effect), all else being equal? Explain briefly, using the output in Figure 16.

My answer: The variable `dereg01` is 1 when deregulation is in effect and 0 otherwise, so its slope is the effect of deregulation vs. no deregulation. This slope is *negative*, so trucking costs are *lower* when deregulation is in effect vs. not, all else being equal.

(d) (1 mark) Figure 17 contains a second regression. How is this regression different from the one in Figure 16?

My answer: The two variables `shipment` and `pctload` have been removed.

(e) (3 marks) Which regression do you prefer, the one in Figure 16 or the one in Figure 17? Explain briefly, giving *two* reasons for your answer, one of which is based on P-values.

My answer: There are two ways of comparing regressions: informally, via R-squared (or adjusted R-squared, which is relevant here since we are comparing regressions with different numbers of x's), or more formally via P-values.

From Figure 16 to Figure 17, R-squared has gone down a lot, from about 60% to about 42%. Equivalently, adjusted R-squared is almost equal to the unadjusted R-squared, and it has gone down too. (Going down a lot doesn't matter for the adjusted R-squared; the fact that it has gone down at all is enough.)

Looking at P-values, we have to be a little careful, because we have removed *two* variables, and the t-tests in Figure 15 properly apply to only one variable at a time. So the best answer is that both of the variables that we removed are significant at the 0.05 level, and therefore removing even one of them is a mistake. (After we remove one, the second one could conceivably become non-significant, though it is hard to imagine that happening here.) A less good, though I think still acceptable, answer, is that both variables are significant, so we shouldn't remove them. (This gets away from the one-at-a-time thing that the t tests are all about, which is why it is not quite as good an answer.) In R, we would do this with `anova`: fit the two models, and feed them both into `anova`, smaller first. That would tell us if we are justified in removing *both* variables. In SAS, it's more complicated, involving `test` lines that I didn't talk about in class, so I didn't think it was fair to throw that at you, even with explanation.

The reason that the adjusted and unadjusted R-squareds are almost the same is that we have a lot of data, 448 observations.

(f) (2 marks) What additional analysis would you like to do before you present your preferred regression model to the president of this trucking company? How will this improve your presentation? Explain briefly.

My answer: Some kind of model checking. I think the most obvious answer is to obtain residual plots, of residuals against fitted values, and, if you want to be diligent, of residuals against each of the explanatory variables. This is the usual thing: if you see any patterns, that is an indication of a problem with your preferred model (eg. curved relationships, if you see curves). If you don't see any patterns, your preferred model is appropriate.

I think residual plots is the best answer, but you can also look for outliers, or check the residuals for normality (which covers the same ground in a bit more detail). Something sensible by way of checking that your preferred model is appropriate, and why that will help, is what I'm after. You don't need to go overboard, since this is only two marks.

Back in the early 1980s, there was a lot of edgy, satirical comedy on British TV. (This was the time of Monty Python and Fawlty Towers, and a little later The Young Ones.) Talking about trucking makes me think of this one, from the same era:

`https://www.youtube.com/watch?v=w9lmCpIzhFo`

Yes, that *is* Mr Bean, in a very different guise.

8. In Question 2, we looked at an analysis of variance of blueberry yields. Unfortunately, the farmer provided us the data in the form shown in Figure 18.

   (a) (2 marks) Explain briefly how the data in Figure 18 are not "tidy" as we defined it in class.

   My answer: The four columns all measure the same thing (blueberry yields), so there should be *one* column of yields and one column identifying varieties. Or, there are two variables, yield and variety, so there should be two columns. Or, there are 32 separate plants, each of which produced one independent measurement, so there should be 32 rows. Any of those.

   (b) (3 marks) Give code, using one or more functions from the `tidyverse`, that will produce a tidy data frame `blueberry`. As input, use the data frame that was read in from the data file (Figure 18). This data frame is called `blueberry`.
   You can assume that `library(tidyverse)` has already been run.

   My answer: There are several columns that are all measurements of the same thing, so this calls for `gather`. The obvious way is this, in one line (plus another to show the result, which you don't need):

```
R> blueberry %>% gather(variety,yield,Berkeley:Sierra)

    variety yield
1  Berkeley  5.13
2  Berkeley  5.36
3  Berkeley  5.20
4  Berkeley  5.15
5  Berkeley  4.96
6  Berkeley  5.14
7  Berkeley  5.54
8  Berkeley  5.22
9      Duke  5.31
10     Duke  4.89
11     Duke  5.09
12     Duke  5.57
13     Duke  5.36
14     Duke  4.71
15     Duke  5.13
16     Duke  5.30
17   Jersey  5.20
18   Jersey  4.92
19   Jersey  5.44
20   Jersey  5.20
21   Jersey  5.17
22   Jersey  5.24
23   Jersey  5.08
24   Jersey  5.13
25   Sierra  5.08
26   Sierra  5.30
27   Sierra  5.43
28   Sierra  4.99
29   Sierra  4.89
30   Sierra  5.30
31   Sierra  5.35
32   Sierra  5.26
```

This is the data frame that, in Question 2, I ran `aov` and `TukeyHSD` on. Without doing the tidying, I wouldn't have been able to run these.

This was actually a pretty easy three marks, since you only had to do one thing (`gather`: no `separate` or `spread` or anything like that).

9. 2017: this question is about detecting multivariate outliers, which we didn't do this year.

Some data were collected on life expectancies in 38 countries. (The life expectancy is the number of years a baby born in the year of the survey can be expected to live.) Specifically, the variables were:

- `lifeexp` overall life expectancy
- `logpertv` logarithm of number of people per TV set in the country
- `logperdr` logarithm of number of people per family doctor in the country
- `lifeexpf` life expectancy for females
- `lifeexpm` life expectancy for males

The logarithms were taken because the original variables were very right-skewed. The data, with these five variables plus the `country` as text, are stored in a data frame called `life`.

Figure 19 shows summaries of the variables in the data set, and correlations between all pairs of variables in the data set. I removed the first column in each case; it contains the names of the countries, so there is no point in including it in the numerical summaries.

Figure 20 shows some R code to calculate leverages.

(a) (3 marks) Why would I want to calculate leverages for this data set? What would leverages tell me that boxplots (or five-number summaries) would not? Explain briefly.

My answer: I want to find observations that are unusual on some variables or a combination of variables. A boxplot will not find observations on a combination of variables (the observation might not be unusual on any one variable by itself). But such an observation *will* have a high leverage.

"I want to find outliers" or "I want to find potential errors" also works. But you need to say what leverages do that boxplots don't for the third point.

(b) (2 marks) What is the reason for calculating the variable called `cutoff` in Figure 20? What is the reason for the numbers 5 and 38 being used in the calculation? Explain briefly.

My answer: This is calculating how large a leverage has to be in order to be considered unusually large. The formula is $2(k+1)/n$ where k is the number of variables (here 5) and n is the number of observations (here 38 countries).

(c) (2 marks) Explain in words what the last line of code in Figure 20, with the two "pipes" `%>%`, does.

My answer: This takes the original data frame, adds a new variable called `lev` containing the leverages, and then picks out the rows where the leverage is bigger than the cutoff $2(k+1)/n$ (and shows all the variables in the result).

(d) (2 marks) What specifally is it about Ethiopia that makes it appear at the end of Figure 20?

My answer: Ethiopia (and the other two countries) have high leverage, so are unusual in some way.

The first thing is to check whether Ethiopia is unusual on any of the variables singly:

- life expectancy: the smallest
- female and male life expectancy: the lowest
- log-people per doctor: the highest

- log-people per TV: very high, though not the highest

Ethiopia is unusual on *all* of those.

(e) (2 marks) What is it about Sudan's `logperdr` and `logpertv` variable values that makes it appear at the end of Figure 20? I am looking for a specific thing for the second mark.

My answer:

- log-people per doctor: above the 3rd quartile, but not the biggest
- log-people per TV: the *smallest* of all.

What makes this especially unusual is the *combination*: these variables have a positive correlation of about 0.56, so you'd expect a country that is high on one to be high on the other, and Sudan is anything but.

In summary: the combination "low on `pertv`" and "high on `perdr`" is unusual, because these variables are *positively* correlated overall.

Perhaps this is clearer if I plot `logperdr` against `logpertv`, and label Sudan so that you can see where it is. First I have to read in the data and create the log-variables:

```
R> life0=read.table("lifeexp.txt",header=T)
R> suppressMessages(library(dplyr))
R> life0 %>% mutate(logperdr=log(perdr), logpertv=log(pertv)) %>%
R>   select(-c(perdr,pertv)) -> life
R> attach(life)
```

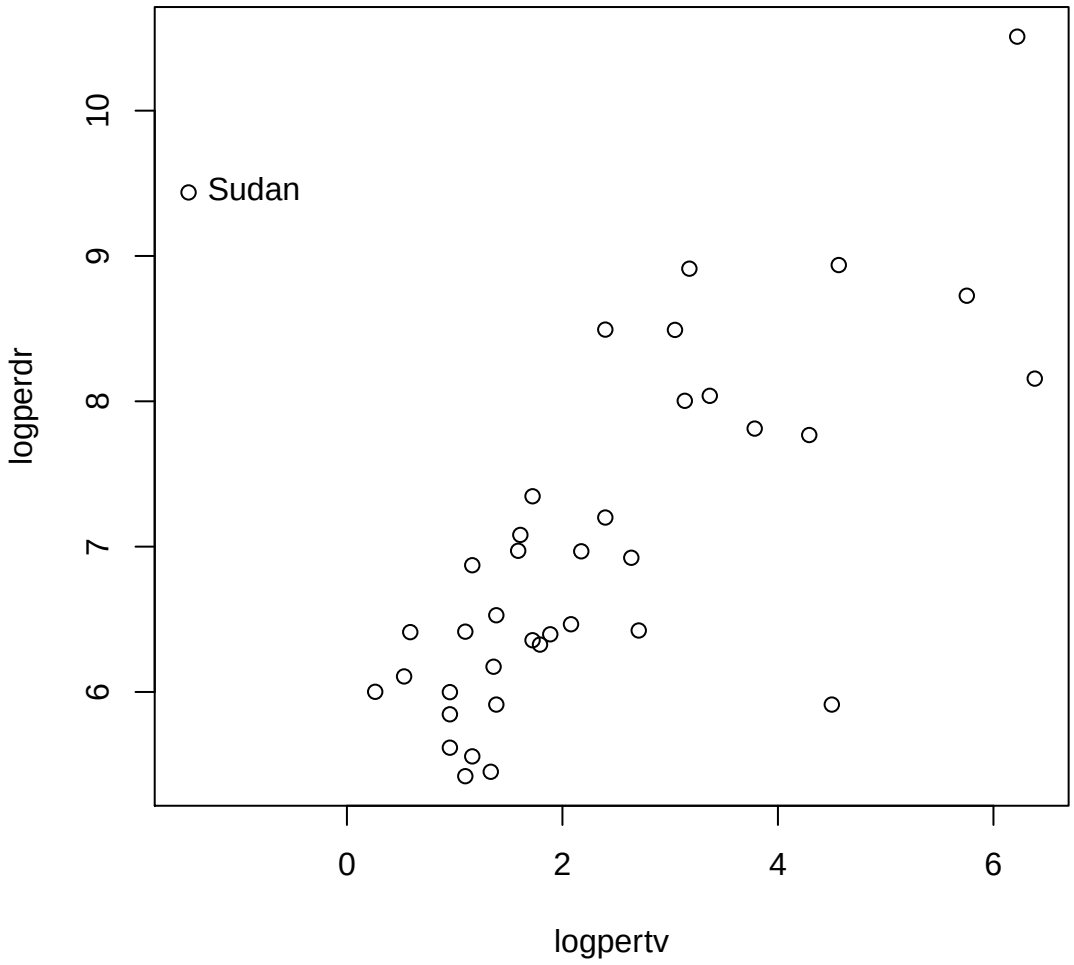
Next, find out which country is Sudan in the data frame, by creating a new variable `i` that is row number, and then picking out the country (we hope there is only one!) whose name is Sudan:

```
R> life %>% mutate(i=row_number()) %>% filter(country=="Sudan")

  country lifeexp lifeexpf lifeexpm logperdr  logpertv  i
1   Sudan      53       54       52 9.437476 -1.469676 30
```

Country 30. Now we plot the two variables against each other, adding some text to label Sudan:

```
R> plot(logperdr~logpertv)
R> text(logpertv[30],logperdr[30],country[30],pos=4)
```



Sudan is the most top-left of all the points. *That's* what makes it unusual: it is a very unusual combination of values on these two variables.

It seems very odd that Sudan should have the *lowest* number of people per TV in the entire data. You'd expect an African country to have a *high* number of people per TV. I think the value in the data is an error, and calculating the leverages has helped us find it.

10. (7 marks) A study was conducted to evaluate the performance of a diesel engine run on three different types of fuel. The response variable was called the Mass Burning Rate, and it was thought to depend on both the Fuel type and on the Brake Power. The data are shown in Figure 21.

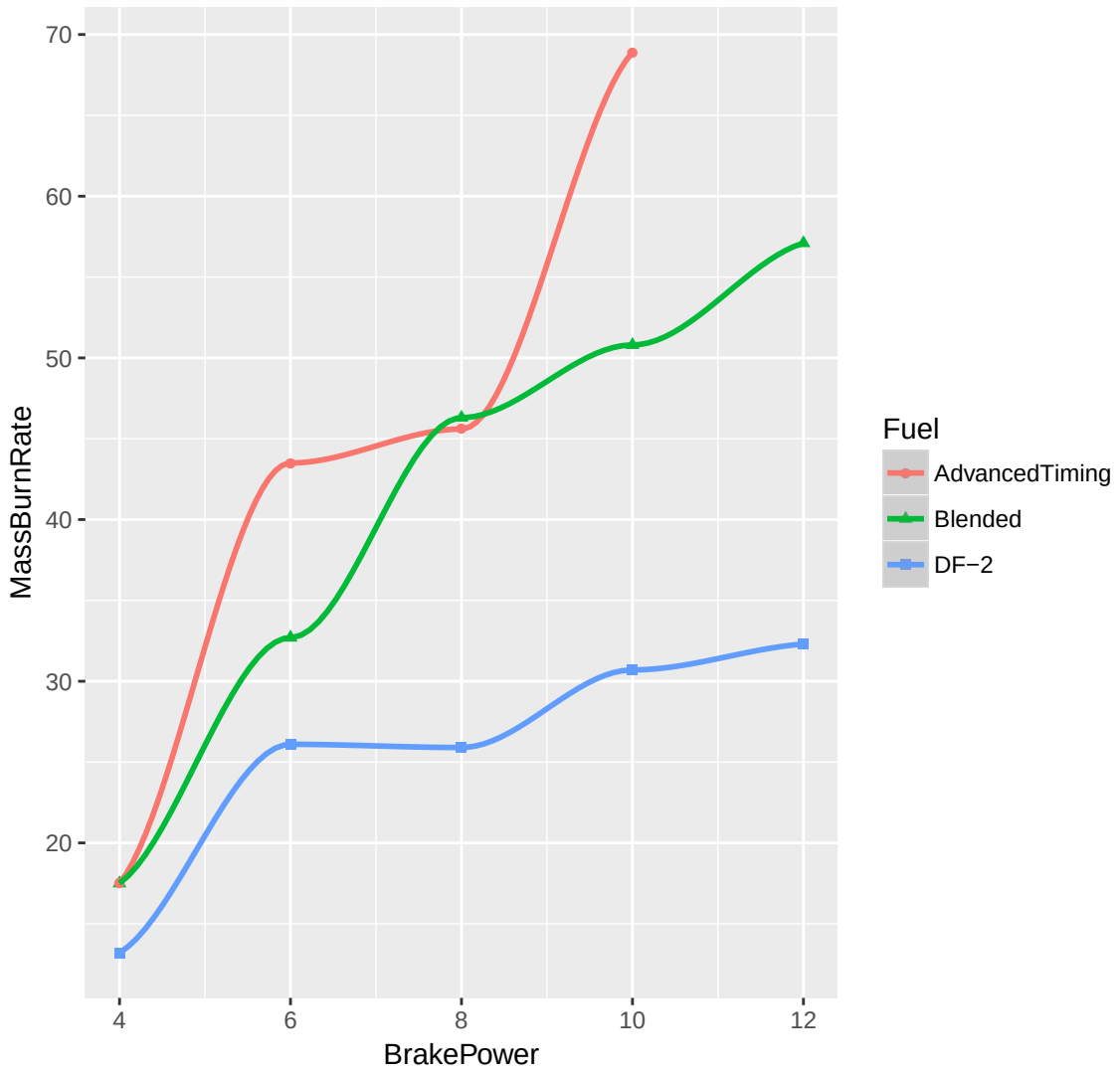
    The data have been read into a data frame called `synfuels`.

    Give R code to create a plot of these data according to the following specifications: there should be a scatterplot of Mass Burn Rate against Brake Power, with the points being different colours and different plotting characters according to the Fuel. There should be a legend enabling the reader to tell which points on the plot correspond to which Fuel type. In addition, there should be a lowess curve for predicting Mass Burn Rate from Brake Power, (2017) for each fuel, (2015) for all the fuels together, shown in brown. Finally, the plot should have the title "Mass Burn Rate for different Fuels", in larger than normal red text.

My answer: 2017: some of this is from the final lecture of the semester, and might go beyond what we do there. Make a call for yourself about how much of this is likely to appear on your final exam.

```
R> synfuels=read_table("synfuels.txt")
R> synfuels
R> ggplot(synfuels,aes(x=BrakePower,y=MassBurnRate,colour=Fuel,pch=Fuel))+
R>   geom_point()+geom_smooth()+
R>   ggtitle("Mass Burn Rate for Different Fuels")+
R>   theme(plot.title=element_text(colour="red",size=22))

Parsed with column specification:
cols(
  BrakePower = col_integer(),
  Fuel = col_character(),
  MassBurnRate = col_double()
)
# A tibble: 14 x 3
   BrakePower           Fuel MassBurnRate
        <int>          <chr>        <dbl>
 1          4           DF-2         13.2
 2          4        Blended         17.5
 3          4 AdvancedTiming         17.5
 4          6           DF-2         26.1
 5          6        Blended         32.7
 6          6 AdvancedTiming         43.5
 7          8           DF-2         25.9
 8          8        Blended         46.3
 9          8 AdvancedTiming         45.6
10         10           DF-2         30.7
11         10        Blended         50.8
12         10 AdvancedTiming         68.9
13         12           DF-2         32.3
14         12        Blended         57.1
`geom_smooth()` using method = 'loess'
There were 30 warnings (use warnings() to see them)
```



The `theme` bit is definitely beyond where we went in 2017. Note that the legend comes for free in `ggplot`, whereas in 2015 (see below) you had to produce it for yourself.

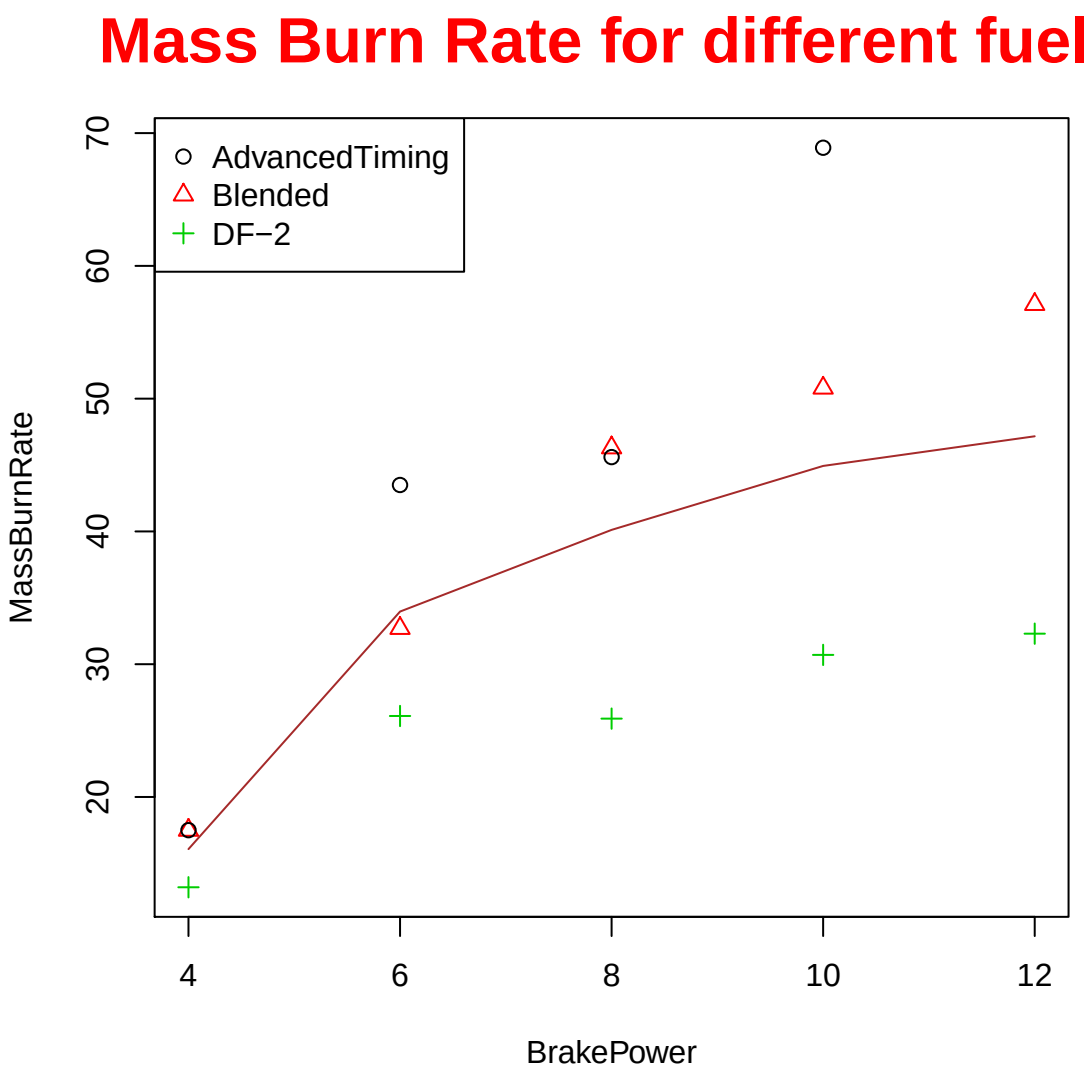
2015: Consult the appropriate pages of your notes to find out how to do all these things. Then the next question is what order to put them in. On the `plot` line goes all the variables to plot, plus the information about the title. Then there is a `lines(lowess)`, and finally a `legend` (the last two of those can be in either order). That gives the code below. First, though, I have to read in the data:

```
R> synfuels=read.table("synfuels.txt",header=T)
R> attach(synfuels)
```

Now the bit that you need:

```
R> plot(MassBurnRate~BrakePower,col=Fuel,pch=as.numeric(Fuel),
R>   main="Mass Burn Rate for different fuels",col.main="red",cex.main=2)
```

```
R> lines(lowess(MassBurnRate~BrakePower),col="brown")
R> fuel.types=levels(Fuel)
R> legend("topleft",legend=fuel.types,col=1:3,pch=1:3)
```



Some comments:

- `pch` requires a number, and `Fuel` is not a number, so you need to do `as.numeric(Fuel)` to get a number.
- To change the title colour and size from the default, you need `col.main` and `cex.main` on the `plot` line. For `cex.main`, any number bigger than 1 is OK.
- `col` is actually an argument to `lines`, not `lowess`, so you specify the colour of the lowess curve as shown.
- The legend needs to show the three different fuel types, which you can get via `levels`.

- There are three different fuel types, so there are three different colours and plotting characters, numbered 1 through 3, that distinguish them.

11. Soldering is a process in which two or more metal items are joined together by melting and flowing a filler metal (solder) into the joint, the filler metal having a lower melting point than the metal being joined. Antimony is sometimes added to tin-lead solder to replace the more expensive tin and to reduce the cost of soldering. The question then is whether adding antimony reduces the strength of the solder joints. An experiment was conducted to assess the strength of solder joints under various combinations of conditions: the amount of antimony added (0, 3, 5 and 10 per cent), and the cooling method: AB: air-blown, FC: furnace-cooled, OQ: oil-quenched, WQ: water-quenched.

    Under each combination of experimental conditions, three solder joints were tested. The measured strengths of these solder joints are labelled `s1, s2, s3` in the data set. The entire data set is shown in Figure 22 in the booklet of code and output.

    Your aim is to give code to extract and/or summarize parts of the data set as described below. You may use either R's `dplyr` tools or SAS to do this, whichever you like, but once you have made your choice you must use that same choice throughout the question. If you choose R's `dplyr` tools you may *not* use `aggregate`. If you choose SAS you need to show how to obtain a new data set (if necessary) satisfying the conditions. (You don't need to give code to list out any new data sets that you create.) Assume that the data have been read into an R data frame or a SAS data set (as appropriate) called `tinlead` with variable names as the column names shown in Figure 22.

    (a) (1 mark) To solve this question, I am going to use **R** / **SAS** (circle one).

My answer: The only way to get this question wrong is not to circle something. Strategy-wise, therefore, circle *something* even if you have no intention of doing this question.

I have to read in the data first. I'm going to do it both ways. It's `read_table2` because the column headings (in mine) don't line up:

```
R> tinlead=read_table2("tinlead2.txt")
R> tinlead

Parsed with column specification:
cols(
  antimony = col_integer(),
  method = col_character(),
  s1 = col_double(),
  s2 = col_double(),
  s3 = col_double()
)
# A tibble: 16 x 5
   antimony method    s1    s2    s3
      <int>  <chr> <dbl> <dbl> <dbl>
 1        0     AB  18.3  19.8  22.9
 2        0     FC  19.4  19.8  20.3
 3        0     OQ  20.0  24.3  21.9
 4        0     WQ  17.6  19.5  18.3
 5        3     AB  21.7  22.9  22.1
 6        3     FC  19.0  20.9  19.9
 7        3     OQ  20.0  20.9  20.4
 8        3     WQ  18.6  19.5  19.0
 9        5     AB  22.9  19.7  21.6
10        5     FC  19.6  16.4  20.5
11        5     OQ  20.9  22.9  20.6
12        5     WQ  22.3  19.5  20.5
13       10     AB  15.8  17.3  17.1
14       10     FC  16.4  17.6  17.6
```

Question 11 continues... This page: _____ of possible 1 points.

```
15       10     OQ  16.4  19.0  18.1
16       10     WQ  15.2  17.1  16.6

SAS> data tinlead;
SAS>   infile '/home/ken/tinlead2.txt' firstobs=2;
SAS>   input antimony method $ s1 s2 s3;
SAS>
SAS> proc print;

Obs    antimony    method     s1      s2      s3
  1        0         AB      18.3    19.8    22.9
  2        0         FC      19.4    19.8    20.3
  3        0         OQ      20.0    24.3    21.9
  4        0         WQ      17.6    19.5    18.3
  5        3         AB      21.7    22.9    22.1
  6        3         FC      19.0    20.9    19.9
  7        3         OQ      20.0    20.9    20.4
  8        3         WQ      18.6    19.5    19.0
  9        5         AB      22.9    19.7    21.6
 10        5         FC      19.6    16.4    20.5
 11        5         OQ      20.9    22.9    20.6
 12        5         WQ      22.3    19.5    20.5
 13       10         AB      15.8    17.3    17.1
 14       10         FC      16.4    17.6    17.6
 15       10         OQ      16.4    19.0    18.1
 16       10         WQ      15.2    17.1    16.6
```

(b) (2 marks) Two of the variables are antimony percents and the strength measurements from the first solder joint. Show only these two variables for all the observations.

My answer: In `dplyr` this is `select`:

```
R> tinlead %>% select(antimony,s1)

# A tibble: 16 x 2
   antimony    s1
      <int> <dbl>
 1        0  18.3
 2        0  19.4
 3        0  20.0
 4        0  17.6
 5        3  21.7
 6        3  19.0
 7        3  20.0
 8        3  18.6
 9        5  22.9
10        5  19.6
11        5  20.9
12        5  22.3
13       10  15.8
14       10  16.4
15       10  16.4
16       10  15.2
```

In SAS you have to `keep` both. Here and below, I've given the `proc print` so that I can check it worked, but you don't need to. The data set you create in each case can have any name, even the same one used over again:

```
SAS> data tinlead2;
SAS>   set tinlead;
SAS>   keep antimony s1;
SAS>
SAS> proc print;

Obs    antimony     s1
  1        0       18.3
  2        0       19.4
  3        0       20.0
  4        0       17.6
  5        3       21.7
  6        3       19.0
  7        3       20.0
  8        3       18.6
  9        5       22.9
 10        5       19.6
 11        5       20.9
 12        5       22.3
 13       10       15.8
 14       10       16.4
 15       10       16.4
 16       10       15.2
```

(c) (2 marks) Show all of the variables *except* for cooling method. Do this *without* naming all the other variables. (Show all of the observations.)

My answer: In R, `select` with a "negative" column:

```
R> tinlead %>% select(-method)

# A tibble: 16 x 4
   antimony    s1    s2    s3
      <int> <dbl> <dbl> <dbl>
 1        0  18.3  19.8  22.9
 2        0  19.4  19.8  20.3
 3        0  20.0  24.3  21.9
 4        0  17.6  19.5  18.3
 5        3  21.7  22.9  22.1
 6        3  19.0  20.9  19.9
 7        3  20.0  20.9  20.4
 8        3  18.6  19.5  19.0
 9        5  22.9  19.7  21.6
10        5  19.6  16.4  20.5
11        5  20.9  22.9  20.6
12        5  22.3  19.5  20.5
13       10  15.8  17.3  17.1
14       10  16.4  17.6  17.6
```

```
15       10  16.4  19.0  18.1
16       10  15.2  17.1  16.6
```

In SAS, `drop`:

```
SAS> data tinlead3;
SAS>   set tinlead;
SAS>   drop method;
SAS>
SAS> proc print;

Obs    antimony     s1      s2      s3
  1        0       18.3    19.8    22.9
  2        0       19.4    19.8    20.3
  3        0       20.0    24.3    21.9
  4        0       17.6    19.5    18.3
  5        3       21.7    22.9    22.1
  6        3       19.0    20.9    19.9
  7        3       20.0    20.9    20.4
  8        3       18.6    19.5    19.0
  9        5       22.9    19.7    21.6
 10        5       19.6    16.4    20.5
 11        5       20.9    22.9    20.6
 12        5       22.3    19.5    20.5
 13       10       15.8    17.3    17.1
 14       10       16.4    17.6    17.6
 15       10       16.4    19.0    18.1
 16       10       15.2    17.1    16.6
```

(d) (2 marks) Show all of the variables for the joints that were air-blown.

My answer: In R, `filter`:

```
R> tinlead %>% filter(method=="AB")

# A tibble: 4 x 5
  antimony method    s1    s2    s3
     <int>  <chr> <dbl> <dbl> <dbl>
1        0     AB  18.3  19.8  22.9
2        3     AB  21.7  22.9  22.1
3        5     AB  22.9  19.7  21.6
4       10     AB  15.8  17.3  17.1
```

In SAS, add a logical condition:

```
SAS> data tinlead4;
SAS>   set tinlead;
SAS>   if method='AB';
SAS>
SAS> proc print;

Obs    antimony    method     s1      s2      s3
 1         0         AB      18.3    19.8    22.9
```

```
 2         3         AB      21.7    22.9    22.1
 3         5         AB      22.9    19.7    21.6
 4        10         AB      15.8    17.3    17.1
```

Make sure to get the right number of equals signs!

(e) (3 marks) Show the variable `s2` for the joints that were water-quenched and had antimony greater than 4 (percent).

My answer: In R, you can add multiple conditions to `filter`, or you can use one `filter` after another. You can use a chain, or temporary variables.

```
R> tinlead %>% filter(method=="WQ",antimony>4)  %>% select(s2)

# A tibble: 2 x 1
     s2
  <dbl>
1  19.5
2  17.1
```

or

```
R> tinlead %>% filter(method=="WQ") %>% filter(antimony>4) %>% select(s2)

# A tibble: 2 x 1
     s2
  <dbl>
1  19.5
2  17.1
```

This one, the `select` has to be at the end, since otherwise you are removing the variables that you are `filter`ing by. SAS has no such qualms:

```
SAS> data tinlead5;
SAS>   set tinlead;
SAS>   if method='WQ';
SAS>   if antimony>4;
SAS>   keep s2;
SAS>
SAS> proc print;

Obs     s2
 1     19.5
 2     17.1
```

(f) (3 marks) Find the mean value of `s3` for each method. (In SAS, assume that you are doing all the parts of this question in sequence.)

My answer: In R, `aggregate` is out, so we have to fall back on `group_by` and `summarize`:

```
R> tinlead %>% group_by(method) %>% summarize(m=mean(s3))

# A tibble: 4 x 2
  method      m
```

```
   <chr>  <dbl>
1     AB 20.925
2     FC 19.575
3     OQ 20.250
4     WQ 18.600
```

In SAS, this is a piece of cake: a straightforward application of `proc means`:

```
SAS> proc means data=tinlead;
SAS>   var s3;
SAS>   class method;

The MEANS Procedure
                              Analysis Variable : s3
               N
method      Obs     N            Mean         Std Dev         Minimum         Maximum
-------------------------------------------------------------------------------------
AB            4     4      20.9250000       2.6056029      17.1000000      22.9000000
FC            4     4      19.5750000       1.3400871      17.6000000      20.5000000
OQ            4     4      20.2500000       1.5800844      18.1000000      21.9000000
WQ            4     4      18.6000000       1.6186414      16.6000000      20.5000000
-------------------------------------------------------------------------------------
```

I insist on your naming the dataset, since `tinlead` is (if we are doing these things in sequence) certainly not the most recently created dataset.

(g) (4 marks) Find the mean value of `s3` for each method, but only for those observations where the antimony percentage is greater than 1.

My answer: The most complicated one of the lot. In R, collect the observations you want, then group-by, then summarize:

```
R> tinlead %>% filter(antimony>1) %>% group_by(method) %>%
R>   summarize(m=mean(s3))

# A tibble: 4 x 2
  method        m
   <chr>    <dbl>
1     AB 20.26667
2     FC 19.33333
3     OQ 19.70000
4     WQ 18.70000
```

In SAS, make a new data set containing the observations with the right antimony, then run `proc means` on that:

```
SAS> data tinlead7;
SAS>   set tinlead;
SAS>   if antimony>1;
SAS>
SAS> proc means;
SAS>   var s3;
SAS>   class method;
```

```
The MEANS Procedure
                              Analysis Variable : s3
               N
method      Obs      N            Mean         Std Dev         Minimum         Maximum
-------------------------------------------------------------------------------------------
AB            3      3      20.2666667       2.7537853      17.1000000      22.1000000
FC            3      3      19.3333333       1.5307950      17.6000000      20.5000000
OQ            3      3      19.7000000       1.3892444      18.1000000      20.6000000
WQ            3      3      18.7000000       1.9672316      16.6000000      20.5000000
-------------------------------------------------------------------------------------------
```

This time you don't need to name the dataset with `data=`, since you just created it.
Or, use `where`, which is simpler:

```
SAS> proc means data=tinlead;
SAS>   where antimony>1;
SAS>   var s3;
SAS>   class method;

The MEANS Procedure
                              Analysis Variable : s3
               N
method      Obs      N            Mean         Std Dev         Minimum         Maximum
-------------------------------------------------------------------------------------------
AB            3      3      20.2666667       2.7537853      17.1000000      22.1000000
FC            3      3      19.3333333       1.5307950      17.6000000      20.5000000
OQ            3      3      19.7000000       1.3892444      18.1000000      20.6000000
WQ            3      3      18.7000000       1.9672316      16.6000000      20.5000000
-------------------------------------------------------------------------------------------
```