Answers and notes

NB In all cases, 'straight' quotes should be used i.e. " rather than " and ' rather than '

I think that I've fixed all of these in this document, but not that you will get an error in R if you copy and paste an example with a 'smart' quote.

Task

1. Check the anscombe data exists:

   Instructions for loading the datasets library are given in the original document

2. The functions mean(), var() and sd() respectively report the arithmetic mean, variance and sample standard deviation, e.g.
3. Use these commands to test all of the components of the Anscombe data. How similar are the x variables? How similar are the y variables?

   Variables are similar but not identical; note that x1 to x3 are the same.

```
For full exploration, use summary()

summary(anscombe)
       x1             x2             x3             x4           y1        
 Min.   : 4.0   Min.   : 4.0   Min.   : 4.0   Min.   : 8   Min.   : 4.260  
 1st Qu.: 6.5   1st Qu.: 6.5   1st Qu.: 6.5   1st Qu.: 8   1st Qu.: 6.315  
 Median : 9.0   Median : 9.0   Median : 9.0   Median : 8   Median : 7.580  
 Mean   : 9.0   Mean   : 9.0   Mean   : 9.0   Mean   : 9   Mean   : 7.501  
 3rd Qu.:11.5   3rd Qu.:11.5   3rd Qu.:11.5   3rd Qu.: 8   3rd Qu.: 8.570  
 Max.   :14.0   Max.   :14.0   Max.   :14.0   Max.   :19   Max.   :10.840  
       y2               y3              y4        
 Min.   :3.100   Min.   : 5.39   Min.   : 5.250  
 1st Qu.:6.695   1st Qu.: 6.25   1st Qu.: 6.170  
 Median :8.140   Median : 7.11   Median : 7.040  
 Mean   :7.501   Mean   : 7.50   Mean   : 7.501  
 3rd Qu.:8.950   3rd Qu.: 7.98   3rd Qu.: 8.190  
 Max.   :9.260   Max.   :12.74   Max.   :12.500  
```

4. Try adding an xlim:

```
> plot(x=anscombe$x1,y=anscombe$y1,xlim=c(1,16),ylim(1,16))
```

5. Now, you might note that the axis labels aren't very friendly. We can add parameters to the plot for axis labels, a main label and a subtitle. All of these take strings. Use main, xlab and ylab to set labels, e.g. plot(…. main="Anscombe plot")

I've used 'x' and 'y' as labels below; the Anscombe data are just x.y pairs. Normally we'd want to state units etc

```
plot(x=anscombe$x1,y=anscombe$y1,xlim=c(1,16),ylim(1,16),xlab=”x”,yl
ab=”y”,main=”Anscombe Quartet [1]”)
```

6. You might also say that the axis tick marks aren't ideal either. We can adjust these using xaxp and yaxp; both require a vector giving the extreme values (i.e. first and last) and the number of gaps between tick labels.

   Update your plot command to use xaxp and yaxp. What are the end points you will use? Do the gaps nest nicely?

   For example:

```
plot(x=anscombe$x1,y=anscombe$y1,xlim=c(1,16),ylim=c(1,16),xla
b="x",ylab="y",main="Anscombe Quartet
[1]",xaxp=c(0,15,5),yaxp=c(0,15,5))
```

For xaxp and yaxp, high-low should divide neatly by gap

7. Update your plot adding in, to start with, pch=16 and cex=1. What effect do these have? Try altering cex: what are the range of values you think it might take? Try different settings for pch; do you have a preferred one.

```
e.g.
plot(x=anscombe$x1,y=anscombe$y1,xlim=c(1,16),ylim=c(1,16),xla
b="x",ylab="y",main="Anscombe Quartet [1]",xaxp=c(0,15,3),
yaxp=c(0,15,3),pch=16,cex=0.5)
```

   Note that cex can be fractional 0-1; cex cannot be negative

   Can you think of a case where alternating symbols may be useful?

   If your data is in a single vector, are structured so that we get repeating rows of different sources (e.g. male,female,m,f, m,f, m,f etc) then it might make sense to recycle symbols. T would be better to restructure though!

8. Choose a colour, and set your markers to that colour.

```
e.g.
plot(x=anscombe$x1,y=anscombe$y1,xlim=c(1,16),ylim=c(1,16),xla
b="x",ylab="y",main="Anscombe Quartet [1]",xaxp=c(0,15,3),
yaxp=c(0,15,3),col="blue")
```

9. Produce a basic sequence:
10. Now, plot x and sin(x) (or a similar trig function of your choice)

11. Now, add in a graph *type*:

```
Using an integer sequence
x <- -5:5
plot(x,sin(x))
```

Then, try

```
plot(x,sin(x),type='l')
plot(x,sin(x),type='h')
plot(x,sin(x),type='b')
```

12. Try the different types. What effect do they have? The original graph as shown may seem a bit coarsely drawn. Use the seq() function (look this up, or refer to last week's notes) to generate a sequence that is finer grain than -10:10. What effect does this have on the graph?

e.g.

```
> x <- seq(-10,10,0.1)
> plot(x,sin(x),type='l')

This gives a much smoother line
```

13-15 are instructions…

13. Repeat the Anscombe graph you produced earlier. You should find that the arrow keys on your keyboard will allow you to step back to a previous command, rather than have to type it.
14. Run the following command (if you have used one of the other Anscombe data sets, then substitute the appropriate column names).
15. Add a line to the plot using:

```
> abline(lm(anscombe$y1~anscombe$x1))
```

   a. Try adding a col= statement to the abline command.

```
e.g. abline(lm(anscombe$y1~anscombe$x1),col="grey")
```

   b. Look at the documentation for graphical parameters:

```
e.g. abline(lm(anscombe$y1~anscombe$x1),col="grey",lty="dotted")
```

16. Download the boxoffice file from moodle.
17. Use the R commands getwd(),setwd() and dir() to navigate to the directory where you have saved this file.
    Important to make sure you use / in path in Windows
18. Read the data in using:

```
> boxoffice <- read.csv('boxoffice-data.csv',header=TRUE,as.is=TRUE,sep=',')
```

19. You should now do some explorations of the data, using commands described in the lecture and in supporting documents. Note that the films are ranked from 1-100. The table will not be re-ordered, unless you proactively take action to do so. This is different to an SQL
    a. Show a list of the top 10 films

```
> boxoffice[1:10,] # note that we have to say which columns we want,
using a comma and empty value means 'all columns'
```

    b. Show a list of film rank,title and adjusted lifetime gross, in reverse order, from 20 to 11

```
> boxoffice[20:11,1:3] # these are cols 1 to 3
```

    c. Create a logical index (see last week's practical notes if needed) based on the boxoffice table, which is TRUE when the value of 'Year' is prior to 1960. Remember, you can refer to columns within a data frame using the dollar notation: boxoffice$Year

```
> logicalInd <- boxoffice$Year < 1960
```

    d. Use that logical index to list the details of the films in the all time list that were released prior to 1960

```
> boxoffice[logicalInd,]
```

    e. Use the notes from week 2 on pattern selection using grep to select films for which the title:
        - Matches the pattern "Star Wars" (anywhere in the title)

```
> strInd <- grep(pattern="Star Wars",x=boxoffice$Title)
> boxoffice[strInd,]
```

        - Has a number at the end of the title (requires knowledge of regular expressions!]

```
> strInd <- grep(pattern="[0-9]$",x=boxoffice$Title)
> boxoffice[strInd,]
```

[0-9] matches a digit; $ matches the end of the string

    f. Use the sum() function to calculate the total Rank value of all films you found in (d)

```
> sum(boxoffice$Rank[logicalInd]) # we extract just boxoffice$Rank,
then subselect rows
```

g. Follow the same processes as used in steps (d) and (e) to find the total Rank of films in the top 200 released in the 1970s.

```
> logicalInd <- (boxoffice$Year >= 1970 & boxoffice$Year <= 1979)
> boxoffice[logicalInd,] # check films selected
> sum(boxoffice$Rank[logicalInd])
```

h. Try calculating the sum of adjusted gross for the top 5 films. What happens?

```
> boxoffice[1:5,] # check films selected
> sum(boxoffice$Adj..Lifetime.Gross[1:5])
```

We get an error:

```
Error in sum(boxoffice$Adj..Lifetime.Gross[1:5]) :
  invalid 'type' (character) of argument
```

20. Using gsub() create a column in boxoffice with a numeric representation of the adjusted gross values

```
boxoffice$adj_usd <
as.numeric(gsub(pattern="[$,]",replacement="",x=boxoffice$Adj..Li
fetime.Gross))
```

21. Create another column with numeric representation of the (unadjusted) lifetime gross

```
boxoffice$gross_usd <
as.numeric(gsub(pattern="[$,]",replacement="",x=boxoffice$Lifetime.G
ross))
```

22. Can you create a column called 'decade', which gives, as text, the decade in which the film was released (e.g. "1970s")?

Hint: for a typical solution you will need to extract a numeric value from boxoffice$Year, such that 1970, 1971, 1972 … 1979, all give you the result '1970'. How will you do this?

You will then need to convert that number to a string, and add on the character 's' at the end.

The operators and functions needed to do this were described in (separate sections of) last week's practical

```
> boxoffice$decade <- paste(boxoffice$Year - boxoffice$Year %% 10,"s",sep="")
```

We subtract year%%10 from year – i.e. the remainder when we divide by 10. So – 1970 has remainder 0, 1971 has remainder 1, and so on. Thus for the 1970s we get 1970-0, 1971-1, 1972-2 etc – all give the answer '1970'; we paste an 's' on to the end.
paste() will coerce the numeric value to a string.

ggplot() in this section, but let us start off with plot()

23. How you arrange a plot to show the rate at which gross declines from highest to lowest (within the top 200)? What variables are needed on the x and y axes?

```
> plot(x=boxoffice$Rank,y=boxoffice$adj)
```

24. Are the labels for gross as you would expect? A simple approach would be show the grosses in millions of dollars rather than in dollars.

```
> plot(x=boxoffice$Rank,y=boxoffice$adj/1000000)
```

We can do a mathematical operation on a vector as we plot it, so this is a simple approach

25. As described in the lecture, we must set this up.
    a. If you have not yet installed tidyverse then do so now (comment on Moodle if you have any problems installing this):
    ```
    > install.packages("tidyverse")
    ```
    b. Load the library:
    ```
    > library(tidyverse)
    ```

The ggplot() function is described in R for Data Science; see: https://r4ds.had.co.nz/data-visualisation.html (NB in the printed edition, ggplot is introduced in chapter 1; in the online edition this material is in chapter 3).

26. We will start by returning to the Anscombe data for a moment.

A ggplot example is given in the question

We will now plot some of the film grosses data using ggplot

27. Use ggplot to produce a plot with lifetime gross as the y variable, and year released as the x variable. As decscribed above, you will need to use a numeric version of lifetime gross.

```
ggplot(boxoffice)
+geom_point(mapping=aes(x=Year,y=gross_usd))
```

Assumes gross_usd has been created from the lifetime gross as in the gsub example above

28. Use ggplot to produce a plot of adjusted gross against rank

```
ggplot(boxoffice)
+geom_point(mapping=aes(x=Rank,y=adj_usd))
```

29. Use ggplot to produce a plot of adjusted gross against year.

```
ggplot(boxoffice)
+geom_point(mapping=aes(x=Year,y=adj_usd))
```

In each case, you should use appropriate functons to set suitable labels.

The last technqiue we shall look at today is the use markers to show further data. The ggplot function allows us to adjust characteristics such as marker size or colour on the basis of a variable.

30. Add the parameter 'colour' to your mapping, for example:

Compare the results with these two plots:

```
> ggplot(boxoffice)
+geom_point(mapping=aes(x=Rank,y=adj_usd/1000000,color=decade))

> ggplot(boxoffice)
+geom_point(mapping=aes(x=Rank,y=adj_usd/1000000,color=Year))
```

How does this change the appearance? Can you explain the differences in the presentation of the legend?

Year is numeric so uses a shading ramp, decades is categorical so uses distinct colour swatches