Part A

1. Download the two data sets `expr.csv` and `biol.csv` from

   `http://biosun01.biostat.jhsph.edu/~kbroman/teaching/statcomp/Rproblems`

2. Use `read.table()`, with appropriate values for the arguments `header` and `sep`, to load the data into objects `expr` and `biol`.

3. Take a look at the data. Note that some values are missing (-99 and .). Re-load the data using `read.table` with the argument `na.strings` set so that these values become NA.

4. Are the datasets `expr` and `biol` data frames, lists or matrices? Use the functions `is.matrix`, `is.list` and `is.data.frame` to find out.

5. Some of the common problems that occur when using `read.table` are that the lines in the file have different numbers of fields (in which case `read.table` bombs) and that some columns are turned into factors without you knowing it.

   Use `lapply` or `sapply` with `mode`, `is.factor` and `is.numeric` to see what the different columns of `expr` and `biol` were turned into.

6. Play around with indexing data frames.

```
names(expr); dimnames(expr); names(biol)

biol[,-1];  biol[1:20,]; biol$f1; biol[,"f2"]

expr[1:10,-c(3,5,6)]; expr$gene3
expr[expr[,3] > 4.5,]
expr[!is.na(expr[,3]) & expr[,3] > 4.5,]
```

7. Use `apply`, `any` and `is.na` to find rows in `biol` with at least one NA.

8. Use `apply` or `sapply` to find the means of each column of `expr`. Note that you may need to use the `na.rm` argument for the function `mean`.

9. Perhaps it would be nicer if the `sample` column for `expr` was made the names of the rows. Use `dimnames` and `as.character` to do that, and then delete the `sample` column. Get the column means again. Also get the SDs and ranges for each column.

10. Calculate the correlation matrix of `expr` using the function `cor`. You'll need to use the argument `use`: try `use="complete.obs"` and `use="pairwise.complete.obs"`.

11. For each row, subtract the average of the first two columns from each of the other four columns, and then delete first two columns. (We probably need to do something special to account for the missing values, but ignore that for now.) Re-calculate the correlation matrix.

12. Use `pairs` to get a scatterplot matrix of the columns of `expr` against each other.

13. Note that the `sample` column in `biol` is like the `sample` column in `expr`, but with an added character in front. Also things are in a different order.

    Remove the character from the front of the sample, and create a new column (a factor), `f3`, containing that character. (Use the function `substring`.) Make the row names of `biol` the sample number (with the character removed), and then get the rows of `biol` in the same order as the rows of `expr`.

14. Use `apply` and `tapply` to find the mean and SD of each column of `expr` within each group `biol$f1==1` and `biol$f1==2`.

15. Use `t.test` (you may need to first do `library(c.test)`) with `apply` and `split` to do t-tests comparing the groups defined by `biol$f2` for the values in each column of `expr`.

Part B

1. Download the data set `mites.txt` from the previously-mentioned webpage.

2. Load the data into an object `mites` using `read.table` (again, using appropriate values for the arguments `header` and `sep`).

3. Use `table` to determine the number of replicates at each dose.

4. We wish to examine whether, at a particular dose, the mites are dying independently with a constant probability.

   (a) Use `tapply` to calculate, for each dose, $\hat{p}$ = overall proportion of dead mites.

   (b) Use `split` to break up `n.dead` into a list, separating the different doses.

   (c) Use `sapply` to calculate, for each dose, the generalized Pearson χ^2 statistic

   $$X^2 = \sum_i \frac{(x_i - 10\hat{p})^2}{10\hat{p}(1-\hat{p})}$$

   (d) Use `pchisq` to get P-values.

   (e) Plot the sample variance, at the different doses, of the numbers of dead mites within each group of ten, against the overall proportion dead at each dose. Add a curve at the expected (binomial) variance.

5. Ignoring the results above (which clearly showed that the mites weren't dying independently), use the function `glm` to fit a generalized linear model with the binomial family and a logit link.