Introduction to R
Session 1: Reading in Data

Before you begin:

- Install RStudio from http://www.rstudio.com/ide/download/ - use the standard installation.

- Go to the course website; http://faculty.washington.edu/kenrice/rintro/ - and locate the datasets we will be using.

1. Ensure you can read in the datasets used in the lecture, and that you obtain the same data summaries seen in the slides.

2. Read in the `crab` data, available on the course site. This dataset describe 173 female horseshoe crabs, and contains data on their weight (in kg), width of shell (in cm), color, condition of their spines, and the number of 'satellites' they have – satellites are male crabs, who attach to the females.

    a. What is the mean number of satellites, in female crabs whose width is <26cm, versus those with width >= 26 cm? (Hint: use the View window to help you)
    b. Tabulate the crabs color, for
        i. All the crabs
        ii. Those with width < 26cm
        iii. Those with width >=26cm?

    Hint: use the `table()` command

3. The `othercrab` data is also on the course site; in contains the same `crab` data, but as a comma-separated file, and has a .csv extension. Read in this dataset and check it is the same values as the version in Q2.

Introduction to R
Session 2: More data summary & using functions

1. Read in the `crab` data in these two ways;
    a. Directly from the command line, using a downloaded copy of the data.
    b. Directly from the course website. (NB make sure you log on to the wireless network first)

2. Cross-tabulate the color of the crabs and their number of satellites. Which colors of female crabs appear to attract more male horseshoe crabs?

3. In the `salary` data, we saw that 4 observations had missing `$salary` variables. Using `is.na()` and other commands, determine which number rows have these missing values.

4. Using the help page for `read.csv()` to help you, read in the `othercrab` data again. (NB the .csv format is widely-used, for small and medium-sized rectangular datasets. It's a good simple choice if you need to transfer data between systems, e.g. from an Excel spreadsheet to R.)

Introduction to R
Session 3: Plotting functions and formulas

1. Using the `crab` data (for the last time!);

    a. Plot number of satellites versus width
    b. Plot number of satellites versus weight, and color-code the points differently if their width exceeds 26cm
    c. Illustrate the distribution of number of satellites, for crabs of different colors. (There are several ways to do this)

    For all of these, try re-sizing the graph after it is drawn; what happens to the axes?

2. Save one of your plots from Q1 as a graphics file (e.g. PNG, JPEG, PDF) and then place it in an external document (e.g. a PowerPoint slide). Are the axes big enough?

3. The `titanic` dataset, on the course site, contains information on who survived the sinking of the Titanic, amount Males/Females, 1st/2nd/3rd class passengers and Crew, Adults/Children. For each category, `n` denotes the total number, and `prop` denotes the proportion of them who survived.

    Using the `stripchart()` function and the formula syntax, illustrate which categories were most and least likely to survive. (Hint: you will need to use the help page for `stripchart()`, but its syntax follows the commands seen in the slides)

Introduction to R
Session 4: Adding features to plots

These exercises use the `mtcars` dataset, that is supplied with R. Use `?mtcars` to access a description of it.

1. Create a figure with boxplots to illustrate the distribution of `mpg` (miles per gallon) for the different `cyl` (number of cylinder) types, where each boxplot in the figure has a different color. Add a legend to the plot.

2. Create a scatterplot to illustrate the relationship of `mpg` (miles per gallon) versus `wt` (car weight).

    a. Using `abline()`, add a straight line that you think summarizes this relationship. (Hint: experiment with a few straight lines, replotting the graph each time)

    b. Add scatterplot smoothers to the plot using the `lowess()` and `supsmu()` functions, where one of the lines is red and the other is blue. (Hint: use the commands in the slides from session 4 on scatterplot smoothers!)

    c. Use `legend()` to add an appropriate legend to the scatterplot.

3. Do question 2 again, but this time include the two scatterplot smoothers that have different line types and different colors, where one line is dotted and the other is dashed. Include an appropriate legend. (Hint: for plotting different line types, use the `lty` option with the `lines()` function, e.g., set `lty=2` in the function call).

4. Put your results from Q1/2/3 into one figure, using `layout()`.

Introduction to R
Session 5: Over and over

1. Using the code from the slides, obtain the approximate value of:

    a. The expected value of the median of a sample of size 501, from the Exp(1) distribution
    b. The variance of the same random variable

    In both cases, compare your answers to those in the slides, that use samples of size 51.

2. The correlation of two random variables can be obtained using the `cor()` function, so for example in the built-in `mtcars` dataset, the correlation of weight and miles per gallon can be obtained from

    ```
    with(mtcars, cor(mpg, wt))
    ```

    Using the correlation as the test statistic, implement a permutation test of the null hypothesis that `mpg` and `wt` are uncorrelated.

Introduction to R
Session 6: More loops, control structures, and bootstrapping

Download the `lawschool` dataset, from the course website, and read it into your R session. This dataset is a random sample of 15 law schools, taken from a collection of 82 participating law schools in a large study of admission practices. Two measurements were made on the entering classes of each school: LSAT, the average score for the class on a national law test, and GPA, the average undergraduate grade-point average for the class.

We will estimate the correlation of `LSAT` and `GPA`, and obtain a 95% confidence interval for this correlation.

1. Illustrate the relationship of `LSAT` and `GPA` using a plot (or plots) of your choice.

2. Calculate the correlation of `LSAT` and `GPA` with the `cor()` function.

3. Bootstrap the correlation of `LSAT` and `GPA` using a `repeat()or while()` loop. Start with 50 bootstrap replicates, and then try using 200, 800, and 3200 bootstrap replicates. Keep the output from all four versions.

4. In one graphical window, plot histograms of the bootstrap correlation values for the four bootstrap replicate sizes considered in the previous question (50, 200, 800, and 3,200 replicates). Also calculate the standard deviation of the correlation of `LSAT` and `GPA` for each of the bootstrap replicate sizes.

5. Calculate a 95% confidence interval for the correlation of `LSAT` and `GPA` using the bootstrap quantiles with 3,200 bootstrap replicates.

Introduction to R
Session 7: Fitting models

1. Using the built-in `mtcars` dataset:

    a. Implement a t-test of the null hypothesis that the average miles per gallon is equal in automatic and manual cars. (The `am` variable is 1 for manual, 0 for automatic) Check that your output is sensible by plotting a boxplot of the same data.

    b. Implement analysis of variance, to assess whether the mean miles per gallon is equal in cars with different numbers of forward gears. Again, check your output using a boxplot, and be careful to use a factor representation of the `gear` variable.

2. Again using the `mtcars` dataset, implement linear regression of miles per gallon on weight. How does this compare to your "eyeball" estimate in Session 3? Obtain a p-value assessing the hypothesis that the linear trend in this dataset is flat – how does it compare to the permutation p-value from Session 5?

3. Obtain a 95% confidence interval for the linear trend between LSAT and GPA, in the `lawschool` dataset. Compare this with what you got in Session 6.

4. [For keen people!] The `titaniclong` dataset on the course site contains individual Titanic survival data – each row of the dataset represents one person. Implement logistic regression of survival (1/0) on class, and produce 95% confidence intervals for the odds ratios comparing survival in 1st and 2nd classes to 3rd class.

Introduction to R
Session 8: Introduction to R Packages

The `ggplot2` package is a widely-used data visualization package for enhanced graphics in R. In this session we will use `ggplot2` for plotting data from the salary dataset.

The two main "generic" plotting functions of `ggplot2` are `qplot()` and `ggplot()`. These functions work by trying to 'guess' a useful form of plot, depending on the user-supplied data and desired geometry. In addition to their help pages, detailed information about these (and other) plotting functions in this package can be found at http://ggplot2.org/ .

1. Using either the drop-down menus or the command line (with the `install.packages()` and `library()` commands) install this package and load it into your current R session.

2. Some `qplot()` commands to plot smooth density curves of salary for each gender are given below. Create a similar density plot of salary by rank.

```
qplot(salary, geom="density", fill=gender, data=salary, xlab="Monthly
Salary", ylab="Density")

qplot(salary, geom="density", fill=gender, data=salary, xlab="Monthly
Salary", ylab="Density", alpha=0.5)
```

3. By adjusting the `geom` argument – described in the documentation for `gplot()` – use `qplot()` to plot a histogram of salaries, and a histogram of salaries across rank. Finally, create a histogram of salary by gender within each rank (hint: use the `interaction()` command).

4. The `ggplot()` command extends the basic idea of qplot; the user has to specify 'aesthetic mappings' (via the `aes()` command) that describe how variables in the data are mapped to visual properties, after which other aspects can be added. For example, for a boxplot of salary by rank gender, we can use

```
ggplot(data=salary, aes(x = rank, y = salary, fill = gender) ) +
geom_boxplot()
```

   Another 'add-on' to the plot it to break it break it down by another variable, using `+facet_wrap(~variablename)`. Using the help page for `facet_wrap()` to help you, create boxplots of salary by gender within each rank across fields.

5. [For keen people!] Illustrate the distribution of salary by gender within each rank for the different starting years.

Introduction to R
Special Exercise

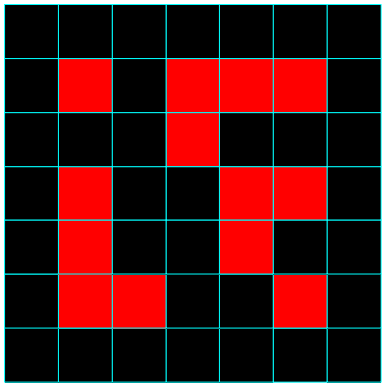
This exercise is for you to try in the evening. We will review various solutions to it in the final session – so even if you don't get a complete solution, tackling some of it will be useful preparation for the final session.

The goal is to implement Conway's Game of Life in R. This 'game' is a simple evolutionary model, where cells on a grid either 'live' or 'die' according to the following rules;

1. Any live cell with fewer than two live neighbors dies, as if caused by under-population.
2. Any live cell with two or three live neighbors lives on to the next generation.
3. Any live cell with more than three live neighbors dies, as if by overcrowding.
4. Any dead cell with exactly three live neighbors becomes a live cell, as if by reproduction.

At each generation, we count the neighbors, then do all the updates to status (i.e. live/dead). For example;

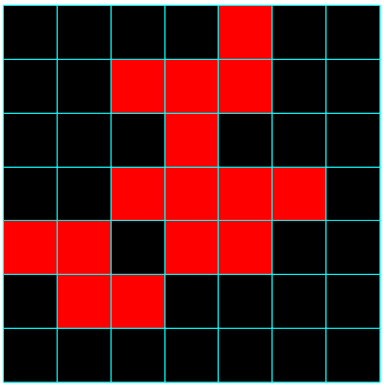
First Generation



Count of Neighbors

1	1	2	2	3	2	1
1	0	3	2	3	1	1
2	2	4	3	6	4	2
2	1	3	3	3	2	1
3	3	4	3	3	4	2
2	2	2	2	2	1	1
1	2	2	1	1	1	1

Second Generation



Your code should plot the grid of alive/dead cells in the R graphics window, for several generations. Some animations showing examples are on the course site.

Some tips: (note we'll give more tips in the evening session)

- Start with a small grid, i.e. 10x10, but write your code so that it can be 'scaled up' later
- Use the `rect()` function to draw the grid
- At each generation, use a double `for()` loop, i.e.

```
for(i in 1:nrows){
   for(j in 1:ncols){
      <do some operations on cell[i,j]>
   }}
```

- There are two options for what to do with the edges and corners;
  - You may count the neighbors based on the visible grid, so e.g. cells on the edges can have 5 neighbors, while those in the corners have up to 3. With this approach, your code for counting neighbors will be different for edge cells, corner cells, and regular internal cells.
  - You may 'wrap around' the definition of neighbors, so e.g. the left edge cells neighbor those on the right edge. With this approach, 'modular arithmetic' will be useful; to do this in R try e.g. `1:20 %% 7`, to generate the 'remainders' when you divide 1:20 by 7.
- Once your code works on small examples, try it on larger grids, initiated at random. For example, `matrix(rbinom(40*40, 1, 0.3),40,40)` generates a 40x40 matrix with entries that have 30% chance of being 0, and 70% chance of being 1.
- [For keen people] Keep track of how long a cell has been alive, and color-code according to age.

Introduction to R
Session 9: Writing functions

1. The *factorial* of a non-negative integer *n* is defined to be

$$n! = n * (n-1) * (n-2) \ldots * 2 * 1,$$

so for example, $4! = 4 * 3 * 2 * 1 = 24$. Create a function that takes a non-negative integer as the argument and returns the factorial of the integer. (Hint: you can use a while loop, but there are many ways to do this). What is the value of 10! ?

2. The formula for converting a temperature in Fahrenheit (F) to Celsius (C) is:

$$C = (5/9) * (F - 32)$$

Write a function that converts a Fahrenheit temperature to Celsius. Use this function to create a data frame containing Fahrenheit values 30, 31, 32, up to 100, and the corresponding temperatures in Celsius.

3. Obtain a root for the following function with the Newton-Raphson method:

$$f(x) = 5x^3 - 4x^2 + 12x - 7$$

(Hint: Implement the Newton-Raphson function given in the session 9 slides).