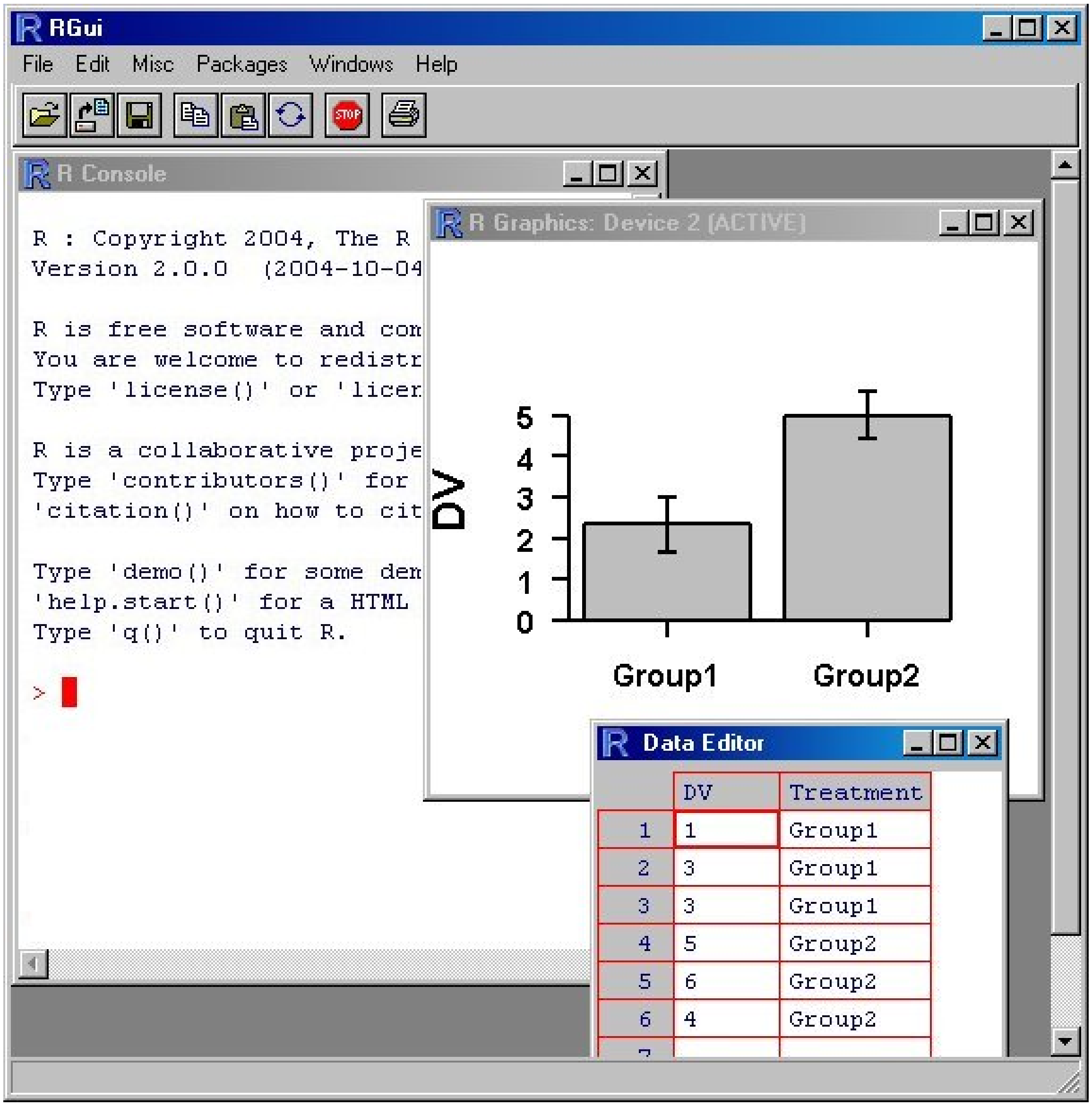
Worksheet 1 - R and Rcmdr

Before beginning this worksheet make sure that you have read the R manual and have either fully installed R (according to the instructions in the R manual) or are running R from the Research Methods CD provided. Other versions of R and Rcmdr obtained elsewhere have not been customized for Bio3011 and therefore do not offer all the required features 'out of the box'.

Demonstration 1 - Running R and Rcmdr

Run R .



```
R : Copyright 2004, The R
Version 2.0.0  (2004-10-04

R is free software and com
You are welcome to redistr
Type 'license()' or 'licen

R is a collaborative proje
Type 'contributors()' for
'citation()' on how to cit

Type 'demo()' for some dem
'help.start()' for a HTML
Type 'q()' to quit R.

>
```

	DV	Treatment
1	1	Group1
2	3	Group1
3	3	Group1
4	5	Group2
5	6	Group2
6	4	Group2

The picture above depicts RGui 2.0.0 running on a Windows system. RGui itself consists of a number of windows (although not all of them are necessarily always on display).

1. R Console - this window is the main R window. It accepts typed R commands and displays R results. When Rcmdr is running, the R Console window is rarely examined or used.
2. R Graphics - this window displays all graphs produced by R (and Rcmdr)
3. Data Editor - this window provides a very crude spreadsheet for entering and modifying data sets.

Note, that both the R Graphics and Data Editor windows are not initially present - they only appear as required.

Everything in R is an object. For example, a single number is an object, a variable is an object, output is an object, a data set is an object, etc. Furthermore, all objects have unique names (that you provide) to enable each object to be referred to. To get the feel of data storage and access in R, try the following

Q1-1. Complete the following table, by **assigning the following entries** (numbers etc) to the corresponding object names and determining the **object class** for each

Name	Entry	Syntax	Class
a	100	hint	
b	Big	hint	
var1	100 & 105 & 110	hint	
var2	5 + 6	hint	
var3	150 to 250	hint	

a. **Print out the contents of the vector you called 'a'**. Notice that the output appears on the line under the syntax that you entered, and that the output is proceeded by a **[1]**. This indicates that the value returned (100) is the first entry in the vector

b. Print out the contents of the vector called 'b'. Again notice that the output is proceeded by **[1]**.

c. Print out the contents of the vector called 'var1'.

d. Print out the contents of the vector called 'var2'. Notice that the output contains the product of the statement rather than the statement itself.

e. Print out the contents of the vector called 'var3'. Notice that the output contains 100 entries (150 to 250) and that it spans multiple lines on the screen. Each new line begins with a number in square brackets **[]** to indicate the index of the entry that immediately follows.

Variables - vectors

Q1-2. Generate the following numeric vectors (variables)

a. The numbers 1, 4, 7 & 9 (call the object y)

b. The numbers 10 to 25 (call the object y1)

c. The sequency of numbers 2, 4, 6, 8...20 (call the object y2)

Q1-3. Generate the following character vectors (factorial/categorical variables)

a. A factor that lists the **sex of individuals as 6 females followed by 6 males**

b. A factor called TEMPERATURE that lists 10 cases of 'High', 10 'Medium & 10 'Low'

c. A factor called TEMPERATURE that lists 'High', 'Medium & 'Low' alternating 10 times

d. A factor called TEMPERATURE that lists 10 cases of 'High', 8 cases of 'Medium' and 11 cases of 'Low'

Q1-4. Print out the contents of the 'TEMPERATURE' factor. A list of factor levels will be printed on the screen. This will take up multiple lines, each of which will start with a number in square brackets **[ ]** to indicate the index number of the entry immediately to the right. At the end, the output also indicates what the names of the factor levels are. These are listed in *alphebetical order*.

Demonstration 2 - Data sets - Data frames(R)

Rarely is only a single biological variable collected. Data are usually collected in sets of variables reflecting tests of relationships, differences between groups, multiple characterizations etc. Consequently, data sets are best organized into collections of variables (vectors). Such collections are called *data frames* in R.

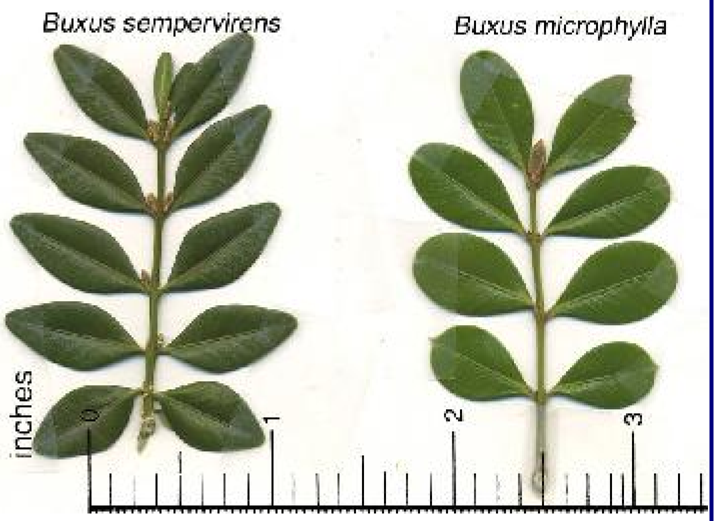
Data frames are generated by combining multiple vectors together whereby each vector becomes a separate column in the data frame. In for a data frame to represent the data properly, the sequence in which observations appear in the vectors (variables) must be the same for each vector and each vector should have the same number of observations. For example, the first observations from each of the vectors to be included in the data frame must represent observations collected from the same sampling unit.

To demonstrate the use of dataframes in R, we will use fictitious data representing the areas of leaves of two species of Japanese Boxwood

Format of the fictitious data set

PLANT	SPECIES	AREA
P1	B.semp	25
P2	B.semp	22
P3	B.semp	29
P4	B.micro	15
P5	B.micro	17
P6	B.micro	20

PLANT	An identifier for each individual plant that was measured (a single leaf was measured from each individual plant)
SPECIES	Categorical listing of whether the individual plant was *Buxus sempervirens* (B.semp) or *Buxus microphyllum* (B.micro)
AREA	The surface area (mm^2) of the leaf measured - Response variable



Q2-1. Lets create the data set in a series of steps. Use the textbox provided in part g below to record the R syntax used in each step

- **a.** First create the categorical (factor) variable containing the listing of B.semp three times and B.micro three times
- **b.** Now create the dependent variable (numeric vector) containing the leaf areas
- **c.** Combine the two variables (vectors) into a single data set (data frame) called LEAVES
- **d.** Print (to the screen) the contents of this new data set called LEAVES
- **e.** You will have noticed that the names of the rows are listed as 1 to 6 (this is the default). In the table above, we can see that there is a variable called PLANT that listed unique plant identification labels. These labels are of no use for any statistics, however, they are useful for identifying particular observations. Consequently it would be good to incorporate these labels as row names in the data set. Create a variable called PLANT that contains a listing of the plant identifications
- **f.** Use this plant identification label variable to define the row names in the data frame called LEAVES
- **g.** In the textbox provided below, list each of the lines of R syntax required to generate the data set

The above syntax forms a list of instructions that R can perform. Such lists are called *scripts*. Scripts offer the following;

- Enable a sequence of tasks such as data entry, analysis and graphical preparation to be repeated quickly and precisely
- Ensure that the sequence of tasks used to complete an analysis are permanently documented
- Simplify performing many similar analyses
- Simplify sharing of data, analyses and techniques

Q2-2. Using the syntax that you entered into the textbox in Q2-1g above, **create and save an R script file**. Make sure you remember where you saved this file - perhaps the desktop is best!

Q2-3. To see how to use a script,

a. **close down R**

b. restart R

c. **Change the working directory (path)** to the location where you saved the script file in Q2-2 above

d. **Source the script file**

Q2-4. There are now at least four objects in the **R workspace**. These should be LEAVES (the data frame - data set), PLANTS (the list of plant ID's), SPECIES (the character vector of plant species) and AREA (the numeric vector of leaf areas).

a. Print (list on screen) the contents of the AREA vector. Note, that this is listing the contents of the AREA vector, this is not the same as asking it to list the contents of the AREA vector within the LEAVES data frame. For example, multiply all of the numbers in the AREA vector by 2. Now print the contents of the AREA vector then the LEAVES data frame. Notice that only the values in the AREA vector have changed - the values within the AREA vector of the LEAVES data frame were not effected.

b. To avoid confusion and clutter, it is therefore always best to **remove single vectors** once a data frame has been created. Remove the PLANTS, SPECIES and AREA vectors.

c. Notice what happens when you now try to access the AREA vector.

d. To access a variable from within a data frame, we use the **$** sign. Print the contents of the LEAVES AREA vector

Q2-5. Since data are stored in vectors, it is possible to access single entries or specific groups of entries. A specific entry is accessed via its **index**. To investigate the range of options, complete the following table.

Access	**Syntax**
print the LEAVES data set	hint
print first leaf area in the LEAVES data set	hint
print the first 3 leaf areas in the LEAVES data	

set		hint
print a list of leaf areas that are greater than 20		hint
print a list of leaf areas for the *B.microphylum* species		hint
print the section of the data set that contains the *B.microphylum* species		hint
alter the second leaf area from 22 to 23		hint

Q2-6. Although it is possible to some data editing this way, for more major editing procedures it is better to either return to Excel or use the **'fix()' function**. Use the 'fix()' function to make a number of changes to the data frame (data set) including adding another column of data (that might represent another variable).

Q2-7. Sometimes it is necessary to **transform** a variable from one scale to another. While it is possible to modify an existing variable (vector), it is safer to create a new variable that contains the altered values. **Examine the use of R for common transformations**. Transform the leaf areas to log (base 10).

Demonstration 3 - Importing data and data files

Although it is possible to generate a data set from scratch using the procedures demonstrated in the above demonstration module, often data sets are better managed with spreadsheet software. R is not designed to be a spreadsheet, and thus, it is necessary to import data into R. We will use the following small data set (in which the feeding metabolic rate of stick insects fed two different diets was recorded)to demonstrate how a data set is imported into R.

Format of the fictitious data set

PHASMID	DIET	MET.RATE
P1	tough	1.25
P2	tough	1.22
P3	tough	1.29
P4	soft	1.51
P5	soft	1.55
P6	soft	1.48

PHASMID	An identifier for each individual stick insect (Phasmid) that was measured
DIET	Categorical listing of whether the food consumed was considered to be tough or soft
MET.RATE	The feeding metabolic rate (mg 0_2/min/g) of phasmids - Response variable

Q3-1. Importing data into R from Excel is a multistage stage process.

a. Enter the above data set into Excel and **save the sheet as a comma delimited text file (CSV)**. Ensure that the column titles (variable names) are in the first row and that you take note where the file is saved. To see the format of this file, open it in Notepad (the windows accessory program). Notice that it is just a straight text file, there is no encryption or encoding.

b. Ensure that the **current working directory is set to the location of this file**

c. **Read (import) the data set into a data table**. Since data exploration and analysis cannot begin until the data is imported into R, the syntax of this step would usually be on the first line in a new script file that is stored with the comma delimited text file version of the data set.

d. To ensure that the data have been successfully imported, print the data frame

Q3-2. As well as importing files, it is often necessary to save a data set (data frame) - particularly if it has been modified and you wish to retain the changes. To demonstrate how to export a data set, we need a data frame (data set) to export. If the LEAVES data frame (from Demonstration 2 above) is no longer present, regenerate the LEAVES data set from above **using the script file** that was generated in Q2-2. To export an R data frame to a text file, you need to **write the data frame to a file**

a. Examine the contents of this comma delimited text file using Notepad

Q3-3. Alternatively, it is also possible to copy and paste data from Excel into R (via the clipboard). Although this method is quicker, there is no record in a R script file as to which Excel file the data originally came from. Furthermore, changes to the Excel data sheet will not be accounted for. **Read the data in from the clipboard**.

Q3-4. Since there is no link between the data and the script when data are imported via the clipboard, it is recommended that the data be stored as a structure within your R script above any commands that use these data. **Place a copy of the data within the R script file that you generated earlier.**.

Demonstration 4 - Generating random numbers and sampling

Randomization is a fundamental concept in sampling design. In order for a sample to truly represent an entire population (all the possible observations), the sample must be collected without bias (intentional or otherwise). Ideally, samples should be collected randomly. For example, the location of quadrats in an area should be determined by a random grid. Individuals from a population to be measured should be selected at random. Likewise the application of treatments should also occur at random.

Given the importance of randomization then, it is necessary to be able to perform randomizations, randomly order treatments and generate random numbers, random coordinates etc.

Random numbers

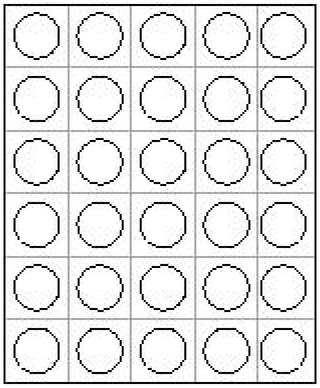
One of the simplest tools to use for random sampling and treatment allocation is a set of **random numbers**. Random numbers are usually given as a fraction (e.g 0.713791167) between 0 and 1. These can be used to generate a series of integer numbers (e.g. 7, 1, 3, 71, 379, 1167 ...) or floating points (e.g. 0.71, 7.13, 71.37, 713.7, 91.167, ...), which in turn can be used to obtain random sampling units (based on individual ID's or random grid coordinates respectively).

Q4-1. For the purpose of demonstration, set the random number generator seed to 1. This will ensure that we all produce the same 'random numbers' and thus will enable you to confirm your answers. **Note that you would not normally want to fix the seed, doing this prevents the numbers being truely random**.

a. Generate a list of random numbers to select 20 individuals at random from a numbered population of 1000 individuals (10 random numbers).

b. Use these to select the location of 10 randomly positioned quadrats within a 100x100m grid.

Random treatment allocation



Lets say that you were designing an experiment in which you intended to investigate the effect of nutrients on the growth rate of a species of plant. You intended to have three different nutrient treatments (high nitrogen, high phosphorus and control) and a total of 10 replicates per treatment. That is, 10 of the pot plants will have a high fertilizer application, 10 will have a high phosphorus application and 10 will have a standard fertilizer application. So, you have a total of 30 seedlings in pots and to assist with watering, you want to place them all on a large table arranged in a matrix with six rows and five columns (see diagram).

To minimize the risks of any bias, and minimize the effects of differences in water, light and temperature in different parts of the table, you decided to arrange the plants on the table randomly.

Q4-2. We will do this in three steps

a. Firstly, generate a character vector that lists each of the treatments 10 times.

b. Use the 'sample()' function to randomly shuffle the ordering of the treatments throughout the vector. Print the vector and notice that this time the arrangement of the treatments is now more randomised.

c. Convert the vector into a matrix with six rows and five columns. Print the matrix.

Random block treatment allocation

Now consider blocking the treatments such that each of the treatments are repeated twice per column, but are randomly allocated within a column.

Q4-3. We will do this in three steps

a. Firstly, generate a character vector that lists each of the treatments twice.

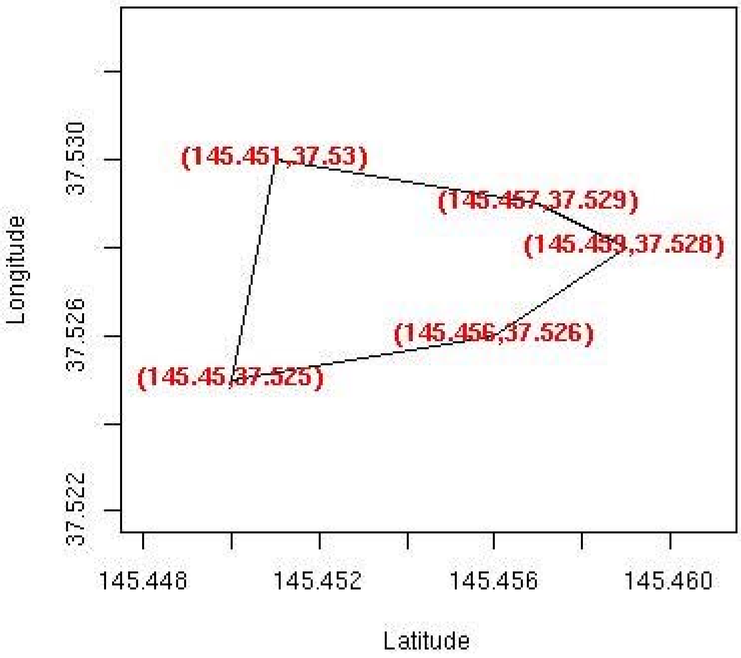
b. Use the 'replicate()' and 'sample()' functions to replicate the shuffling of treatment orders five times (once for each column)

c. Print out the matrix and compare it to the fully randomized matrix from Q4-2 above..

Random map coordinates

Now, lets consider designing an experiment in which a number of point quadrats (lets say three) are to be established in a number of different sites (logged and unlogged) within the Bunyip State Park (Gembrook). These points are to be used for stationary 10 minute bird surveys and you have decided that the location of each of the point quadrats within each site should be determined via random coordinates to minimize sampling bias.

For the purpose of the demonstration, we will illustrate the process of determining the five random coordinate sets (point quadrats) for a single site. The diagram illustrates the 'site' boundaries as well as the boundary grid references (x,y).



Q4-4. We will do this as a series of steps

- Firstly, setup vectors of latitude and longitude coordinates according to the points on the map. Make sure that the first and last coordinates are the same so that the polygon is closed.

a. Combine these into a single matrix of coordinates. Call this matrix 'site'

b. Use these coordinates to plot a map showing the boundaries of the site.

c. From here we need to introduce a extension package called 'sp'. This is a package that contains functions for spatial mapping and sampling. Load the **'sp' package**. HINT. Note that the 'sp' package does not automatically come with R, and may need to be downloaded from the R web page (www.r-preject.org)

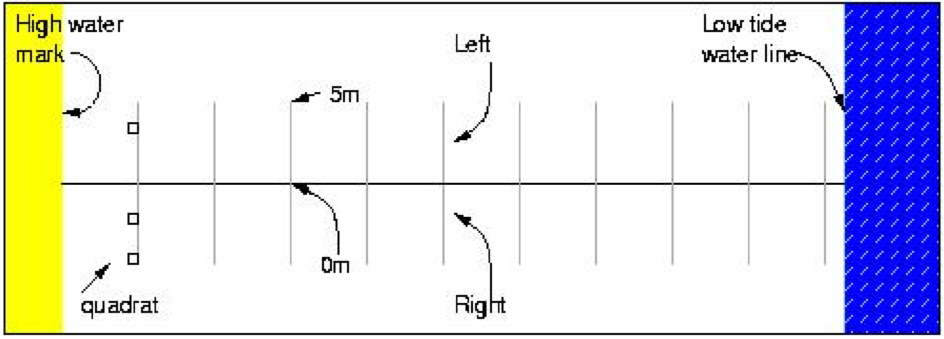
d. Use the 'Polygon()' function that comes with the 'sp' package to create what is called a 'Spatial ring'. This is just a polygon with a few extra parameters.

e. Use the 'spsample()' function that comes with the 'sp' package to generate a list of 5 random coordinates within the 'Spatial ring'. Note that this function is only guaranteed to produce approximately the specified number of random coordinates, and will often produce a couple more or less. Furthermore, some locations might prove to be unsuitable (if for example, the coordinates represented a position in the middle of a lake). Consequently, it is usually best to request a 50% more than are actually required and simply ignore any extras. Print out the list of random coordinates

f. Plot these 5 points on the map created above

Random distances

Lets say you were interested in examining the changes that occur in rocky intertidal invertebrate communities along an exposed rock platform. To do so, you had decided to establish five transects each running perpendicular to the beach and extending from the high water mark to the low tide water line. Every 10 meters along the transects, three 0.5x0.5m quadrats were to be used to count the number and diversity of invertebrate species (see diagram below). To minimize bias, the position of these quadrats with respect the transect were to be at random distances (0 to 5 meters) and directions (either left or right) from the transect line. Therefore, it is was necessary to generate a list of random distances (from 0 to 5 meters) and directions (left or right) to use for the positioning of each quadrat.



Q4-5. Again, we will do this as a series of steps

- **a.** Generate a list of 30 random directions (Left or Right) to represent which side of the transect each quadrat should be placed
- **b.** Generate a list of 30 random distances (between 0 and 5 meters rounded off to two decimal places) to represent the distance (in meters) that each quadrat should be placed from the transect
- **c.** Combine these into a large list of random direction/distance sequences called 'quadrats'.

  Note again that it is usually a good idea to generate 50% more random numbers, sequences (etc) than you expect to need to allow for invalid or inappropriate combinations. For example, in the above situation you may have set an a priori decision not to measure rock pools on the rock platform since they provide very different conditions that may not truly represent the location along the intertidal zone. Consequently, if a random sequence dictates that a quadrat is to be placed in a rock pool, this sequence would be skipped. Likewise, if a random sequence dictated that a quadrat should be placed in a location that overlaps a previously sampled area, it might also be appropriate to skip to the next random sequence.

Welcome to the end of Worksheet 1