**CPSC 260**
Spring 2022

Treat this document like a "Try It Out" from the textbook Beginning R by Gardener. As you read through it, you will learn about list type objects. Italicized text contains tasks for you to perform in R. I provide the code for most, but not all, of these.* After successful completion of a task, copy and paste your R code from the R script file window AND your output from the **Console** window into this Word document. Please use a color other than black. You will upload this Word document to BB when finished.
*Note: R does not like " " copied from Word. Type the commands in yourself to avoid problems. This will also help you learn and understand the material better.

## List Objects

Lists are like atomic vectors, because they group data into a one-dimensional set. However, list elements are actually other R objects. So, a list is a collection of objects placed together inside a single object (the list).

*Take a look at the data set (from the book) called my.list to see an example:*
*my.list*

*Input:*

*my.list*

*Output:*

```
$mow
[1] 12 15 17 11 15

$unmow
[1] 8 9 7 9

$data3
 [1]  6  7  8  7  6  3  8  9 10  7  6  9

$data7
 [1] 23.0 17.0 12.5 11.0 17.0 12.0 14.5
 [8]  9.0 11.0  9.0 12.5 14.5 17.0  8.0
[15] 21.0
```

This example is very simple, containing four vectors of different lengths. So what is the benefit of using a list to hold these four vectors, rather than storing them as columns in a data frame? All vectors must be the same length to create a data frame!

If they are not, but you still want to make a data frame, then you will need to pad out the data with NAs so that columns are the same length.

*To see this for yourself:*

- *Create two vectors of different lengths. Call them vec.short and vec.long:*
  *vec.short<-c(1,2,3,4,5)*
  *vec.long<-c(1,2,3,4,5,6,7,8)*

- *Try to create a data frame using vec.short and vec.long as the columns. [You* *should get an error. Copy the error message into this document.]*

```
Error in data.frame(vec.short, vec.long) : 
  arguments imply differing number of rows: 5, 8
```

- *Pad vec.short with NAs, and then check the vector type:*
  *vec.short<-c(vec.short, NA, NA, NA)*
  *typeof(vec.short)*

*typeof(vec.short)*
*"double"*

- *Try to create a data frame using vec.long and the new vec.short.*

vecs<-data.frame(vec.short,vec.long)

```
vecs

  vec.short vec.long
1         1        1
2         2        2
3         3        3
4         4        4
5         5        5
6        NA        6
7        NA        7
8        NA        8
```

When you look at the data set *my.list*, you see each vector listed separately along with its name, which is prefixed with a dollar sign. Let's learn more about the vectors contained in our list object!

*Check out the structure of my.list:  str(my.list)*

*str(my.list)*

```
List of 4
 $ mow  : int [1:5] 12 15 17 11 15
 $ unmow: int [1:4] 8 9 7 9
 $ data3: num [1:12] 6 7 8 7 6 3 8 9 10 7 ...
 $ data7: num [1:15] 23 17 12.5 11 17 12 14.5 9 11 9 ...
```

A list can contain objects of various types. For example, you might have a matrix, a data frame, and several vectors. The data set *my.list* contains vectors of two different data types: numeric and integer.

**Making a List Type Object**

**CPSC 260**
Spring 2022

If you have several separate objects and want to create a list from them, use the list() command. Note that the names of the objects will not be retained. You will have to add them afterwards.

*Create a list from the following vectors, data frame, and matrix (data sets from the book): data7, habitats, fw, and bird. Look at your result, and then check out its structure.*
*my.list2 = list(data7, habitats, fw, bird)*
*my.list2*
*str(my.list2)*
***Choose your own name for the list object created here.*

```
Testlist

[[1]]
 [1] 23.0 17.0 12.5 11.0 17.0 12.0 14.5  9.0
 [9] 11.0  9.0 12.5 14.5 17.0  8.0 21.0

[[2]]
[1] "Garden"   "Hedgerow" "Parkland" "Pasture"
[5] "Woodland"

[[3]]
         count speed
Taw          9     2
Torridge    25     3
Ouse        15     5
Exe          2     9
Lyn         14    14
Brook       25    24
Ditch       24    29
Fal         47    34

[[4]]
              Garden Hedgerow Parkland Pasture
Blackbird         47       10       40       2
Chaffinch         19        3        5       0
Great Tit         50        0       10       7
House Sparrow     46       16        8       4
Robin              9        3        0       0
Song Thrush        4        0        6       0
              Woodland
Blackbird            2
Chaffinch            2
Great Tit            0
House Sparrow        0
Robin                2
Song Thrush          0


str(testlist) #structure of testlist
List of 4
 $ : num [1:15] 23 17 12.5 11 17 12 14.5 9 11 9 ...
 $ : chr [1:5] "Garden" "Hedgerow" "Parkland" "Pasture" ...
 $ :'data.frame':      8 obs. of  2 variables:
  ..$ count: num [1:8] 9 25 15 2 14 25 24 47
  ..$ speed: num [1:8] 2 3 5 9 14 24 29 34
 $ : num [1:6, 1:5] 47 19 50 46 9 4 10 3 0 16 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:6] "Blackbird" "Chaffinch" "Great Tit" "House Sparrow
" ...
  .. ..$ : chr [1:5] "Garden" "Hedgerow" "Parkland" "Pasture" ...
```

*Add the names of each object to the list. Look at the list again to verify the names have been correctly added.*
*names(my.list2) = c("data7", "habitats" ,"fw", "bird")*
*my.list2*

```
names(testlist)=c("data7","habitats","fw","bird")
> testlist

$data7
 [1] 23.0 17.0 12.5 11.0 17.0 12.0 14.5  9.0
 [9] 11.0  9.0 12.5 14.5 17.0  8.0 21.0

$habitats
[1] "Garden"   "Hedgerow" "Parkland" "Pasture"
[5] "Woodland"

$fw
         count speed
Taw          9     2
Torridge    25     3
Ouse        15     5
Exe          2     9
Lyn         14    14
Brook       25    24
Ditch       24    29
Fal         47    34

$bird
              Garden Hedgerow Parkland Pasture
Blackbird         47       10       40       2
Chaffinch         19        3        5       0
Great Tit         50        0       10       7
House Sparrow     46       16        8       4
Robin              9        3        0       0
Song Thrush        4        0        6       0
              Woodland
Blackbird            2
Chaffinch            2
Great Tit            0
House Sparrow        0
Robin                2
Song Thrush          0
```

## Manipulating List Objects

When you have a list, the square bracket notation gives a different result compared to other data objects. To fully extract an object from the list, you must use double brackets or the dollar sign $ notation.

*Discover what happens when you try using single brackets, double brackets, and $ to extract the third object in the list. Check the class of each result to really see the difference.*
*my.list2[3]*

*my.list2[[3]]*
*my.list2$fw*
***Don't forget to use the name you gave the list you just created rather than my.list2.*

*class( my.list2[3] )*
*class( my.list2[[3]] )*
*class( my.list2$fw )*

*testlist[3] #testing single brackets*
*class(testlist[3]) #checking the class of the result*

*testlist[[3]] #testing double brackets*
*class(testlist[[3]])*

*testlist$fw #testing dollar sign*
*class(testlist$fw)*

```
testlist[3]
$fw
         count speed
Taw          9     2
Torridge    25     3
Ouse        15     5
Exe          2     9
Lyn         14    14
Brook       25    24
Ditch       24    29
Fal         47    34

testlist[[3]] #testing double brackets
         count speed
Taw          9     2
Torridge    25     3
Ouse        15     5
Exe          2     9
Lyn         14    14
Brook       25    24
Ditch       24    29
Fal         47    34

testlist$fw #testing dollar sign
         count speed
Taw          9     2
Torridge    25     3
Ouse        15     5
Exe          2     9
Lyn         14    14
Brook       25    24
Ditch       24    29
Fal         47    34


class(testlist)
[1] "list"
> class(testlist[3])
[1] "list"
> class(testlist[[3]])
[1] "data.frame"
> class(testlist$fw)
```

```
[1] "data.frame"
```

You can also extract parts of objects contained inside the list. I recommend using the $ notation.

*Try extracting just the first column of the data frame contained in your list: my.list2$fw[,1]*

```
testlist$fw[,1] #extracting first column of fw
```

```
[1]  9 25 15  2 14 25 24 47
```

How did that work? Well *mylist2$fw* is a data frame. The $ (and double bracket) notation takes *fw* out of the list, and you may treat it as the data frame that it is. You can now use the standard [row,col] notation for data frames and matrices.

If you ever need to use list objects in your work, then I recommend installing the *rlist* library. It contains a wide variety of functions that can be used to more easily handle list type objects.