rIntermediate

conger

1/20/2016

Learning how to interpret and create for loops.

given the for loop:

```
sp_ids = unique(iris$Species)

output = matrix(0, nrow=length(sp_ids), ncol=ncol(iris)-1)
rownames(output) = sp_ids
colnames(output) = names(iris[ , -ncol(iris)])

for(i in seq_along(sp_ids)) {
    iris_sp = subset(iris, subset=Species == sp_ids[i], select=-Species)
    for(j in 1:(ncol(iris_sp))) {
        x = 0
        y = 0
        if (nrow(iris_sp) > 0) {
            for(k in 1:nrow(iris_sp)) {
                x = x + iris_sp[k, j]
                y = y + 1
            }
            output[i, j] = x / y
        }
    }
}
output
```

1.Describe the values stored in the object output. In other words what did the loops create?

```
matrix(0, nrow=length(sp_ids), ncol=ncol(iris)-1)
```

creates a 3x4 matrix of 0's, a placeholder matrix.

```
rownames(output) = sp_ids
colnames(output) = names(iris[ , -ncol(iris)])
```

names the rows after the iris species and the columns are the values of 4 parameters. the for loop gives the average(arithmetic mean) of all the column values for each iris species.

2. Describe using pseudo-code how output was calculated, for example,
Create a place holder of 0 matrices and label the rows and columns according to their original location in the raw data. counter from i to total number of iris species, use the species type counter j from 1 to the total number of irirs species, start x and y at as 0 placeholders, then if the row of iris species is greater than 0, loop through k from 1 to the total number of iris species through create values x= number of items + number of data for that species and y= total number of counters, output x/y, which is the average of the data.

3. The variables in the loop were named so as to be vague. How can the objects output, x, and y could be renamed such that it is clearer what is occurring in the loop. x=DataSummation y=numberOfDataPoints output=mean_values

written prettier:

```
sp_ids = unique(iris$Species)

mean_values = matrix(0, nrow=length(sp_ids), ncol=ncol(iris)-1)
rownames(output) = sp_ids
colnames(output) = names(iris[ , -ncol(iris)])

for(i in seq_along(sp_ids)) {
    iris_sp = subset(iris, subset=Species == sp_ids[i], select=-Species)
    for(j in 1:(ncol(iris_sp))) {
        dataSummation = 0
        numberOfDataPoints = 0
        if (nrow(iris_sp) > 0) {
            for(k in 1:nrow(iris_sp)) {
                dataSummation = dataSummation + iris_sp[k, j]
                numberOfDataPoints  = numberOfDataPoints  + 1
            }
            mean_values[i, j] = dataSummation / numberOfDataPoints
        }
    }
}
output
```

4. It is possible to accomplish the same task using fewer lines of code? Please suggest one other way to calculate output that decreases the number of loops by 1. Use a preset function that calculates the mean of data without the need for a for loop. use sum().

5.Have a vector x with the numbers 1:10. Write a for loop that will produce a vector y that contains the sum of x up to that index of x. So for example the elements of x are 1, 2, 3, and so on and the elements of y would be 1, 3, 6, and so on.

BOTH CODES WORK:

```
y=0
x=c(1:10)
y[1] = 1
for (i in 2:10) {
   y[i] = x[i]+sum(x[1:i-1])
}
y
```

and

```
y=0
x=c(1:10)
for (i in 0:11) {
   y[i] = x[i]+sum(x[1:i-1])
}
y
[1]  1  3  6 10 15 21 28 36 45 55 NA
```

6. Modify your for loop so that if the sum is greater than 10 the value of y is set to NA

```
x=c(1:10) y=0 for (i in 0:11) { y[i] = x[i]+sum(x[1:i-1]) if (y[i]<=10) { print(y) } else if (y[i]>10) { print ('NA') } } y
```

7. Place your for loop into a function that accepts as its argument any vector of arbitrary length and it will return y.

```
summation_alg = function(vector_length){
  test=NULL
  y=0
  x=c(1:vector_length)
  for (i in 0:vector_length+1) {
   y[i] = x[i]+sum(x[1:vector_length-1])
}
  return(y)
  print('y')
}
summation_alg(5) #let k=5
```