R INTERMEDIATE ASSIGNMENTS: HOMEWORK DUE FRIDAY, JANUARY 22, 2016

```r
data(iris)
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```r
names(iris)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"
## [5] "Species"
```

```r
dim(iris)
```

```
## [1] 150   5
```

```r
sp_ids = unique(iris$Species)
sp_ids
```

```
## [1] setosa     versicolor virginica
## Levels: setosa versicolor virginica
```

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

```
##      [,1] [,2] [,3] [,4]
## [1,]    0    0    0    0
## [2,]    0    0    0    0
## [3,]    0    0    0    0
```

```r
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
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## setosa              5.006       3.428        1.462       0.246
## versicolor          5.936       2.770        4.260       1.326
## virginica           6.588       2.974        5.552       2.026
```

Question 1. The output values are the averages for each trait scored for each species. First the rows of the data set are designated as the species (setosa, versicolor , and virginica), then the columns are all of the columns from the data set, but taking out the species column (-Species). Then taking the average for each trait for each species.

```
data(iris)

versicolor<-iris[iris$Species=="versicolor",]
mean(versicolor$Sepal.Width)
```

```
## [1] 2.77
```

```
setosa<-iris[iris$Species=="setosa",]
mean(setosa$Sepal.Width)
```

```
## [1] 3.428
```

```
virginica<-iris[iris$Species=="virginica",]
mean(virginica$Sepal.Width)
```

```
## [1] 2.974
```

Question 2. Just to double check that this is what the loop in 1 was doing. I subsetted each Species name in the column species to contain it's own vector for each trait. (so each species would be compile its own data frame of 4 columns of trait data)

Here is my pseudo code below with the hashtags.

```
sp_ids = unique(iris$Species)
sp_ids
```

```
## [1] setosa     versicolor virginica
## Levels: setosa versicolor virginica
```

```
#pulling out the names of the species in the species column

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

```
##      [,1] [,2] [,3] [,4]
## [1,]    0    0    0    0
## [2,]    0    0    0    0
## [3,]    0    0    0    0
```

```r
#made a matrix that is the length in rows of the species names, and column
#length of the columns found in the data set - the last column.
rownames(output) = sp_ids #labeled the rows with the species names
colnames(output) = names(iris[ , -ncol(iris)])
#labeled the columns with the traits

for(i in seq_along(sp_ids)) { # loop through sp
  # subset to only specific sp
  iris_sp = subset(iris, subset=Species == sp_ids[i], select=-Species)
    for(j in 1:(ncol(iris_sp))) {
        # compute mean for each column (ie. trait)
        output[i, j] = mean(iris_sp[ , j])
      #i'th species for the j'th trait
    }
}
output
```

```
##            Sepal.Length Sepal.Width Petal.Length Petal.Width
## setosa            5.006       3.428        1.462       0.246
## versicolor        5.936       2.770        4.260       1.326
## virginica         6.588       2.974        5.552       2.026
```

```r
#when you designate the output, i and j are what you are putting as your columns
#and rows. Then when you are outputting this and you want the mean for each
#species for each column of data, you must designate the i, j for it to generate
#the output.
```

Questions 3 and 4: This loops generates the mean values for each species for the Sepal and Petal Length and Width. Here it is much clearer that we are subs-setting the loop to the iris species. The columns are the traits measured. then for each species for each column of data, we want the mean computed. This is straight forward because R actually does the mean function. Previously, (in the other loop) X and Y were designated and created much more R code than necessary.

SUM OF A SEQUENCE

5. Here is my Vector of Y that sums up the index vector of x. I did include 0 in the length of the vector for y.

```r
vectorx<-c(1:10)

y<-0
for(i in 0:11)
  {
  y[i]<-vectorx[i] + sum(vectorx[1:i-1])
  }
  print(y)
```

```
##  [1]  1  3  6 10 15 21 28 36 45 55 NA
```

6. Modified loop that spits out NA for y values greater than 10.

```
vectorx<-c(1:10)
vectorx
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

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

```
## [1] 1
## [1] 1 3
## [1] 1 3 6
## [1]  1  3  6 10
## [1] "NA"
## [1] "NA"
## [1] "NA"
## [1] "NA"
## [1] "NA"
## [1] "NA"
```

7.

Loop Function with X of any length: I answered this question from two approaches. Whether there was a completely new vector someone wanted to put that they came up with with a set length and values for the vector in mind, or if they wanted the computer to generate a random set of values and a random length for the vector. .

```
vec<-vector()
vec<-c(vec, 1:50)

#i personally decided i wanted a vector with the values of 1:50.
#But these numbers can be changed if the person wants them changed.

 anyy<-0
for(i in 1:25) #the length of the values chosen can be changed as well.
  {
  anyy[i]<-vec[i]+sum(vec[1:i-1])
  }
print(anyy)
```

```
##  [1]   1   3   6  10  15  21  28  36  45  55  66  78  91 105 120 136 153
## [18] 171 190 210 231 253 276 300 325
```

This loop allows you to designate a vector of any range you want in the c(vec, 1:x) ***you choose the range of x that you want***. Here i chose the range to be between the numbers 1 to 50. Then in the "for" statement (line 256), you can designate the length of the vector added into the function. here I wanted my vector to consist of the first 25 numbers between 1:50.

But let's say you want to generate a vector that consisted of random numbers generated by the computer (the vector can be of any length that is also randomly generated by the computer) and the generated/ random numbers can be between any set numbers then you would do the following:

For example, I want a random vector of x (let's call it randomx) to be a length of 15 numbers, and I want the computer to generate those numbers between 1:500.

```
length<-sample(1:500,1)
length
```

```
## [1] 363
```

```
#this computes one number randomly chosen between 1:500, which will designate
#a randomized length of the x vector.

randomx<- sample(1:500, length, replace=T)
randomx
```

```
##   [1] 133 309 381 416  16  48 181  37 135 336  15 138  32 214 102 345 282
##  [18] 232 119 449 208 472 372 289 365  73   9 310 381 185 336  77 162 160
##  [35] 209 124  34 241 360 312 310 395  38 242 363 397  76 173 277 181 139
##  [52] 491 291 173 313 335  40 418 402  92  89 369 172 326 397 293 459  72
##  [69] 318 240 325  34 346 402 260  32 302 131 356 255 153 315 352 283 442
##  [86] 368 322 218  44 215 474  21 480 232  90 109 420 421 144 243 326 207
## [103] 331 349 405  91  30 395 370 121 186 420 179 460   6 157 196 480  93
## [120] 317 306 437 434 310 330 135 337 227 405 440 448 466 138 134 298 386
## [137] 219 124 248  48 138  82 310 491 446 425 350  34 490 250 135  30 479
## [154]  17 115  21 456  24   9  44  37 303 133 225 214 341 207  53 280 253
## [171] 182 142 306 247 196 457 236 496 258 394 410 239 278 244 258 492  73
## [188] 281 109 264  33 490  22 416 345 179 109 278 166 169 405  95 254 302
## [205] 103 485 281 409 384 272 357 346 236 415 250 470 197 163 333 120 241
## [222]  61 206  39 191 394 446 411 332 483 134  60 455 336  86 218 245 191
## [239] 210 286 261 269 116  70 229 376  40 108  82 176  93 259 413 337 398
## [256] 435 150  70 420 269 178 371   2 105 245 165 302  43 438 242 486 162
## [273] 460  98 481 227 175 201 167 332  40 142 201  11 200 295 104 410 442
## [290] 124  44 290 303 185  98  98 121 255 334 109  42  62 374 315 354 168
## [307] 465 200 279 336 432 139 238 272   1 467 223 175 194 218  38 352 352
## [324]  10 268  26 440 327 469 492 428 420 307 432 218  32 311 185 398 126
## [341] 123 453 201 225 384  79 128 450 432 492 233 249 225 425 225 105 272
## [358] 236 201 379  60 356 430
```

```
#this is generating random values for  your x vector -- the length of said
#vector is previously randomized in the prior step.

 ry<-0
for(i in 1:15)  #I am just going to designate that I want to choose the first
  #15 numbers generated in my randomized vector sequence for my function.
  #But this sequence can also be randomly generated via the computer using the
  #same function above.
  {
      ry[i]<-randomx[i]+sum(randomx[1:i-1])
}
print(ry)
```

```
##  [1]  133  442  823 1239 1255 1303 1484 1521 1656 1992 2007 2145 2177 2391
## [15] 2493
```

```
#each time this code is run, it will generate a different x and y vector,
#respectively.
```

So here are two loops generated to accept any random vector you personally decide to choose, OR one that will be randomly generated via the computer.

Each time you run this code, a new x vector(randomx) of a random length is generated, giving you a new values for the Y vector (ry).