HW2 R Intermediate

Nick Weber

1/18/2018

```
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
```

```
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
```

```
##            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
```

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

#The object 'output' is a matrix of 3 rows and 4 columns. The rows represent the three d
ifferent species, and the values are averages of Sepal Length, Sepal Width, Petal Lengt
h, and Petal Width for each of the three species.
```

```
#2. Describe using pseudo-code how `output` was calculated.

sp_ids = unique(iris$Species) #Asking for names of different species present in 'iris'.

output = matrix(0, nrow=length(sp_ids), ncol=ncol(iris)-1) #Specifying layout of output
 matrix.
rownames(output) = sp_ids #Assigning row names to the output matrix (aka species names).
colnames(output) = names(iris[ , -ncol(iris)]) #Assigning column names to output matrix
 (aka column names present in 'iris' minus the last column).

for(i in seq_along(sp_ids)) { #seq_along takes a vector (here 'sp_ids') and creates a se
quence up to the count of elements in the vector.
    iris_sp = subset(iris, subset=Species == sp_ids[i], select=-Species) #Subset data fo
r a particular species...
    for(j in 1:(ncol(iris_sp))) { #For each column, for that particular species...
        x = 0
        y = 0
        if (nrow(iris_sp) > 0) { #Start with x and y at 0 if there is at least one row o
f values for that species...
            for(k in 1:nrow(iris_sp)) { #For each column, sum all of the values (x) and
 count the number of values (y)...
                x = x + iris_sp[k, j]
                y = y + 1
            }
            output[i, j] = x / y #Now divide the sum of values for each column by the nu
mber of values in that column.
        }
    }
}
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
```

```
#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.

#The object 'output' could be renamed 'speciesaverages', since it is a matrix of the dif
ferent averages for each of the species. The object 'x' could be renamed 'sumvalues' and
 the object 'y' could be renamed 'countvalues'.
```

```
#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.

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))) {
            output[i, j] = sum(iris_sp[, j]) / nrow(iris_sp)
        }
    }
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
```

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

x <- c(1:10)
y=NULL

for(i in 1:length(x)) {
  y[i] <- sum(1:i)
}
y
```

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

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

x <- c(1:10)
y=NULL

for(i in 1:length(x)) {
  y[i] <- sum(1:i)
  if (y[i] > 10) {
    print("NA")
  }
  else {
    print(y[i])
  }
}
```

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

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

x=NULL
y=NULL

yfunc <- function(x) {
  for(i in 1:length(x)) {
  y[i] <- sum(1:i)
  if (y[i] > 10) {
    print("NA")
  }
  else {
    print(y[i])
  }
  }
}

z <- c(1:20)
yfunc(z)
```

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