Self-Test
SER 50 R Workshop

S Mooney

June 20, 2017

1. Use c to create a vector named myNumbers from these numbers:
   9 9 5 8 4 2 8

   ```
   > myNumbers <- c(9, 9, 5, 8, 4, 2, 8)
   >
   ```

2. Create a vector named myLetters of 26 lowercase letters from a to z. (Hint: check out the *letters* vector is built into R)

   ```
   > myLetters <- letters
   ```

3. Combine the vectors myNumbers and myLetters columnwise into a matrix called my myObj.

   ```
   > myObj <- cbind(myNumbers, myLetters)
   ```

4. Create a vector of numbers named myvec with the numbers from 19 to 10 (i.e. 19, then 18, then 17.)

   ```
   > myvec <- 19:10
   ```

5. Sort myvec in ascending order

   ```
   > myvec <- sort(myvec)
   ```

6. Create a logical vector called teens with the value TRUE for all numbers in myvec that are more than 12 (i.e. your resulting vector should start FALSE FALSE FALSE TRUE TRUE .)

   ```
   > teens <- myvec > 12
   ```

7. Create an object containing one item named 'mynumbers' with values 1,2,3, and 4, and one item named 'myletters' and has values e, f, and g. (Hint: what basic object type should this be?)

   ```
   > list(mynumbers=1:4, myletters=c('e', 'f', 'g'))
   ```

```
$mynumbers
[1] 1 2 3 4

$myletters
[1] "e" "f" "g"
```

8. Run the following code:

```
> data(esoph)
> myarray <- table(esoph$agegp, esoph$alcgp, esoph$tobgp)
```

(a) What value is in the myarray cell representing over age 75, consuming 120+ g of alcohol/day and 30+ g of tobacco/day? (Hint: age 75+ is row 6, 120+ g is column 4, and 30+ g is depth slice 4)

```
> myarray

, ,  = 0-9g/day


        0-39g/day 40-79 80-119 120+
  25-34         1     1      1    1
  35-44         1     1      1    1
  45-54         1     1      1    1
  55-64         1     1      1    1
  65-74         1     1      1    1
  75+           1     1      1    1

, ,  = 10-19


        0-39g/day 40-79 80-119 120+
  25-34         1     1      1    1
  35-44         1     1      1    1
  45-54         1     1      1    1
  55-64         1     1      1    1
  65-74         1     1      1    1
  75+           1     1      1    1

, ,  = 20-29


        0-39g/day 40-79 80-119 120+
  25-34         1     1      0    1
  35-44         1     1      1    1
  45-54         1     1      1    1
  55-64         1     1      1    1
  65-74         1     1      1    1
```

```
  75+            0     1      0    0

, ,  = 30+


        0-39g/day 40-79 80-119 120+
  25-34         1     1      1    1
  35-44         1     1      1    0
  45-54         1     1      1    1
  55-64         1     1      1    1
  65-74         1     0      1    1
  75+           1     1      0    0
```

```
> # 0
```

(b) How would you index myarray to return that value?

```
> myarray[6,4,4]
```

```
[1] 0
```

9. (a) Create a list with two items: a vector with the numbers from 1 to 5 named 'numbers' and a character string with your name named 'name'. Your list should look like:

```
$numbers
[1] 1 2 3 4 5

$name
[1] "Steve"
```

(Except that your name probably isn't Steve)

```
> mylist <- list(numbers=1:5, name="Steve")
> mylist
```

(b) How would you index the number 3?

```
> mylist$numbers[3]
```

```
[1] 3
```

10. Create a logical vector with the value TRUE for every row in the esoph data frame (that you loaded in step 1) for which ncases is greater than zero. Your vector should look like:

```
 [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [8] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE
[15] FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE
[22]  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE
[29] FALSE  TRUE  TRUE FALSE FALSE FALSE  TRUE
[36]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
[43]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```
[50]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
[57]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
[64]  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE
[71]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
[78]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE
[85]  TRUE  TRUE  TRUE  TRUE
```

Hint: You can use the names() function to return the names of the variables in a data frame

```
> data(esoph)
> mylogical <- esoph$ncases > 0
> mylogical
```

11. create a numerical vector with the number of controls for every row in esoph for which there are any cases. Your vector should look like:

```
 [1]  1 14 23 14  3  4 46 38 21 15  7 16 14  5  4
[16]  4  4  3  4 49 22 12  6 40 21 17  6 18 15  6
[31]  4 10  7  3  6 48 14  7 34 10  9 13 12  3  1
[46]  4  2  1  1 18  6  3  5  3  1  1  1  2  1
```

```
> control_count <- esoph$ncontrols[esoph$ncases > 0]
> control_count
```

12. Create a 2x2 matrix named results with 12 in the a cell, 24 in the b cell, 20 in the c cell and 8 in the d cell as follows:

```
> results <- matrix(c(12, 24, 20, 8), nrow=2, ncol=2, byrow=TRUE)
```

(a) Calculate the row and column sums using rowSums() and colSums()

```
> rowSums(results)

[1] 36 28

> colSums(results)

[1] 32 32
```

(b) Calculate the row and column sums using apply()

```
> # Row
> apply(results, 1, sum)

[1] 36 28

> # Column
> apply(results, 2, sum)

[1] 32 32
```

(c) Make a 2-item list named totals with first element row sums and second element col sums

```
> totals <- list(rowSums=rowSums(results), colSums=colSums(results))
```

(d) Make a copy of totals named totals.copy

```
> totals.copy <- totals
```

(e) Use mapply to add the row sums and column sums from totals to the row sums and column sums from totals.copy. Your result should look like:

```
rowSums colSums
    128     128
```

```
> mapply(sum, totals, totals.copy)
```

```
rowSums colSums
    128     128
```

13. First, load the USArrests database built into R using data(USArrests).

```
> data(USArrests)
```

(a) What is the median number of assault arrests per 100,000 people?

```
> median(USArrests$Assault)
```

```
[1] 159
```

(b) What was the murder rate in the state(s) that has (have) the median rate of assault arrests have?

```
> USArrests$Murder[USArrests$Assault == median(USArrests$Assault)]
```

```
[1] 7.4 4.9
```

(c) which states are they?

```
> rownames(USArrests[which(USArrests$Assault == median(USArrests$Assault)),])
```

```
[1] "New Jersey" "Oregon"
```

i. Perform a linear regression predicting the number of murders by the number of assaults.

```
> lm(Murder ~ Assault, data=USArrests)
```

```
Call:
lm(formula = Murder ~ Assault, data = USArrests)

Coefficients:
(Intercept)      Assault
    0.63168      0.04191
```

ii. What is the slope of that regression line?

```
> model <- lm(Murder ~ Assault, data=USArrests)
> summary(model)
```

```
Call:
lm(formula = Murder ~ Assault, data = USArrests)

Residuals:
    Min      1Q  Median      3Q     Max
-4.8528 -1.7456 -0.3979  1.3044  7.9256

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.631683   0.854776   0.739    0.464
Assault     0.041909   0.004507   9.298  2.6e-12

(Intercept)
Assault     ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.629 on 48 degrees of freedom
Multiple R-squared:  0.643,        Adjusted R-squared:  0.6356
F-statistic: 86.45 on 1 and 48 DF,  p-value: 2.596e-12
> # 0.042
```

iii. is it significantly different from 0 at p <0.05?)

```
> summary(model)
Call:
lm(formula = Murder ~ Assault, data = USArrests)

Residuals:
    Min      1Q  Median      3Q     Max
-4.8528 -1.7456 -0.3979  1.3044  7.9256

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.631683   0.854776   0.739    0.464
Assault     0.041909   0.004507   9.298  2.6e-12

(Intercept)
Assault     ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.629 on 48 degrees of freedom
Multiple R-squared:  0.643,        Adjusted R-squared:  0.6356
F-statistic: 86.45 on 1 and 48 DF,  p-value: 2.596e-12
```

```
> # Yes
```

14. Duncan questions

    (a) Load the Duncan dataset, first by loading the car package, then loading the dataset itself:

    ```
    > library(car)
    > data(Duncan)
    > head(Duncan)
               type income education prestige
    accountant prof     62        86       82
    pilot      prof     72        76       83
    architect  prof     75        92       90
    author     prof     55        90       76
    chemist    prof     64        86       90
    minister   prof     21        84       87
    ```

    (b) How many of the jobs in the Duncan dataset are type=prof?

    ```
    > table(Duncan$type)
      bc prof   wc
      21   18    6
    > #18
    ```

    (c) Create a new data frame named prof.jobs that is the subset of the Duncan dataset that is professional jobs.

    ```
    > prof.jobs<-subset(Duncan, Duncan$type=='prof')
    > prof.jobs
                   type income education prestige
    accountant     prof     62        86       82
    pilot          prof     72        76       83
    architect      prof     75        92       90
    author         prof     55        90       76
    chemist        prof     64        86       90
    minister       prof     21        84       87
    professor      prof     64        93       93
    dentist        prof     80       100       90
    engineer       prof     72        86       88
    undertaker     prof     42        74       57
    lawyer         prof     76        98       89
    physician      prof     76        97       97
    welfare.worker prof     41        84       59
    teacher        prof     48        91       73
    contractor     prof     53        45       76
    factory.owner  prof     60        56       81
    store.manager  prof     42        44       45
    banker         prof     78        82       92
    ```

(d) What is the slope of the regression line predicting income from prestige among professional jobs?

```
> x <-lm(income~prestige, data=prof.jobs)
> summary(x)

Call:
lm(formula = income ~ prestige, data = prof.jobs)

Residuals:
    Min      1Q  Median      3Q     Max 
-43.923  -3.146   0.161   8.159  12.849 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.3234    18.1190   0.018  0.98598
prestige      0.7425     0.2220   3.344  0.00412

(Intercept)   
prestige    **
---
Signif. codes:  
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 12.91 on 16 degrees of freedom
Multiple R-squared:  0.4114,	Adjusted R-squared:  0.3746 
F-statistic: 11.18 on 1 and 16 DF,  p-value: 0.004117

> #Slope = 0.743
```

15. Air quality questions

(a) Load the airquality dataset that is built into R

```
> data(airquality)
```

(b) Create a logical variable in the air quality dataset named niceout, which has the value true when the temperature was above 65 and below 80. (You can pick different temperatures if you prefer it to be warmer or colder out)

```
> airquality$niceout <- airquality$Temp >65 & airquality$Temp<80
```

(c) Use logistic regression to compute odds ratios of being nice out by month.

```
> nice <- glm(niceout ~ as.factor(Month), family=binomial(logit), data=airquality)
```

(d) What are the odds of a day in June being nice compared to a day in May?

```
> exp(coef(nice))
```

```
      (Intercept) as.factor(Month)6
       0.93750000        1.39487179
as.factor(Month)7 as.factor(Month)8
       0.07356322        0.50793651
as.factor(Month)9
       1.60000000
```

```
> # The odds of 'nice' day in June is 1.39 times the odds of a nice day in May.
```

(e) Plot the number of nice days in each month (hint: use tapply() to sum the number of nice days per month) Use either basic graphics or ggplot.

```
> nice_by_month <- data.frame(
+     days=tapply(airquality$niceout, airquality$Month, sum))
> barplot(nice_by_month$days,
+           names.arg=c("May", "June", "July", "August", "September"))
```