Lab1 Solution

Vectors

1. Create a vector u that has values -10, -9, -8, . . . ,0. How many different ways can you use?

```
u <- c(-10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0)
u <- -10:0
u <- seq(-10,0)
# In the following two lines, from = and to = can be omitted.
u <- seq(from = -10, to = 0, by = 1)
u <- seq(from = -10, to = 0, length.out = 11)
```

2. Create another vector v that has values -0.1, 0.4, 0.9, 1.4, ., and there are 11 numbers (aka terms) in v. How many different ways can you use?

```
v <- c(-0.1, 0.4, 0.9, 1.4, 1.9, 2.4, 2.9, 3.4, 3.9, 4.4, 4.9)
# In the following line, from = can be omitted.
v <- seq(from = -0.1, by = 0.5, length.out = 11)
print(v)
```

```
##  [1] -0.1  0.4  0.9  1.4  1.9  2.4  2.9  3.4  3.9  4.4  4.9
```

3. Calculate the vector of u+v and u*v.

```
u+v
```

```
##  [1] -10.1  -8.6  -7.1  -5.6  -4.1  -2.6  -1.1   0.4   1.9   3.4   4.9
```

```
u*v
```

```
##  [1]   1.0  -3.6  -7.2  -9.8 -11.4 -12.0 -11.6 -10.2  -7.8  -4.4   0.0
```

4. Increase all terms in u by 1, and then take away 20% from all terms in v.

```
u+1
```

```
##  [1] -9 -8 -7 -6 -5 -4 -3 -2 -1  0  1
```

```
(1-0.2)*v
```

```
##  [1] -0.08  0.32  0.72  1.12  1.52  1.92  2.32  2.72  3.12  3.52  3.92
```

5. Create a vector w that contains all the numbers from u and then v. Report the length of w.

```
w <- c(u,v)
print(w)
```

```
##  [1] -10.0  -9.0  -8.0  -7.0  -6.0  -5.0  -4.0  -3.0  -2.0  -1.0   0.0
## [12]  -0.1   0.4   0.9   1.4   1.9   2.4   2.9   3.4   3.9   4.4   4.9
```

```
length(w)
```

```
## [1] 22
```

6. Use a command to return the 14th, 15th and 16th value of w. What about the 2nd, the 5th, 9th and 21st value of w? What is the 23rd value?

```
w[14:16]
```

```
## [1] 0.9 1.4 1.9
```

```
w[c(2,5,9,21)]
```

```
## [1] -9.0 -6.0 -2.0  4.4
```

```
w[23]    # No error is reported, but a NA
```

```
## [1] NA
```

7. Replace the 3rd term of w by 100. Then replace the 7th, 15th and 22nd terms by 200, 300 and 400 simultaneously.

```
w[3] <- 100
w[c(7,15,22)] <- c(200,300,400)
print(w)
```

```
##  [1] -10.0  -9.0 100.0  -7.0  -6.0  -5.0 200.0  -3.0  -2.0  -1.0   0.0
## [12]  -0.1   0.4   0.9 300.0   1.9   2.4   2.9   3.4   3.9   4.4 400.0
```

8. Remove u.

```
rm(u)
```

9. Remove all the objects in the environment.

```
rm(list = ls())
```

10. Create a vector p of the values of $e^x cos(x)$ at $x = 3, 3.1, 3.2, ..., 6$.

```
x <- seq(3,6,by=0.1)
p <- exp(x)*cos(x)
p
```

```
##  [1] -19.884531 -22.178753 -24.490697 -26.773182 -28.969238 -31.011186
##  [7] -32.819775 -34.303360 -35.357194 -35.862834 -35.687732 -34.685042
## [13] -32.693695 -29.538816 -25.032529 -18.975233 -11.157417  -1.362099
## [19]  10.632038  25.046705  42.099201  61.996630  84.929067 111.061586
## [25] 140.525075 173.405776 209.733494 249.468441 292.486707 338.564378
```

```
## [31] 387.360340
```

11. Find the maximum/minimum value in p and the index (position) of that value in p.

```
max_in_p <- max(p)
min_in_p <- min(p)
index_max <- which.max(p)  #first index
index_min <- which.min(p)  #first index
indices_max <- which(p == max(p)) #all indices of the max value
indices_min <- which(p == min(p)) #all indices of the max value
print(max_in_p)
```

```
## [1] 387.3603
```

```
print(min_in_p)
```

```
## [1] -35.86283
```

```
print(index_max)
```

```
## [1] 31
```

```
print(index_min)
```

```
## [1] 10
```

12. Sort p in the descending order.

```
sort(p, decreasing = T)
```

```
##  [1] 387.360340 338.564378 292.486707 249.468441 209.733494 173.405776
##  [7] 140.525075 111.061586  84.929067  61.996630  42.099201  25.046705
## [13]  10.632038  -1.362099 -11.157417 -18.975233 -19.884531 -22.178753
## [19] -24.490697 -25.032529 -26.773182 -28.969238 -29.538816 -31.011186
## [25] -32.693695 -32.819775 -34.303360 -34.685042 -35.357194 -35.687732
## [31] -35.862834
```

13. Create (4, 6, 3, 4, 6, 3, . . . , 4, 6, 3) where there are 10 occurrences of 4.

```
tmp <- c(4,6,3)
rep(tmp,10)
```

```
##  [1] 4 6 3 4 6 3 4 6 3 4 6 3 4 6 3 4 6 3 4 6 3 4 6 3 4 6 3 4 6 3
```

14. Create (4, 4, . . . , 4, 6, 6, . . . , 6, 3, 3, . . . , 3) where there are 10 occurrences of 4, 20 occurrences of 6 and 30

occurrences of 3.

```
rep(tmp,times = c(10,20,30))
```

```
##  [1] 4 4 4 4 4 4 4 4 4 4 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 3 3 3 3 3
## [36] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
```

Matrices

1. Create the following matrix and assign it to the variable `b_matrix`.

```
b_matrix <- matrix(
  seq(1,39,by=2),4,5,byrow=T,
  dimnames = list(
    c("A", "B", "C", "D"),
    c("a", "b", "c", "d", "e")
  ))
b_matrix
```

```
##    a  b  c  d  e
## A  1  3  5  7  9
## B 11 13 15 17 19
## C 21 23 25 27 29
## D 31 33 35 37 39
```

2. Extract a sub-matrix from `b_matrix` named `subB` as follows. Try to use as many possible ways as you can (positive and negative indices).

```
subB <- b_matrix[c(1,2,4),c(2,3)]
subB <- b_matrix[-c(3),-c(1,4,5)]
subB <- b_matrix[c(1,2,4),-c(1,4,5)]
subB <- b_matrix[-c(3),c(2,3)]
# all the above will result in the same subB
subB
```

```
##    b  c
## A  3  5
## B 13 15
## D 33 35
```

3. In R, `%*%` is an operator for matrix multiplication. Compute `a_matrix %*% b_matrix` and `a_matrix %*% subB`. Discuss the results you get from R.

```
a_matrix <- matrix(
  1:12,
  nrow = 4,             #ncol = 3 works the same
  dimnames = list(
    c("one", "two", "three", "four"),
    c("eins", "zwei", "drei")
  )
)

dim(a_matrix)
```

```
## [1] 4 3
```

```
dim(b_matrix)
```

```
## [1] 4 5
```

```r
dim(subB)
```

```
## [1] 3 2
```

```r
# a_matrix %*% b_matrix
# The above command doesn't work and throws an non-comformable arguments error.
# As the number of columns in a_matrix isn't equal to the number of rows in b_matrix.
# Specifically, ncol(a_matrix) != nrow(b_matrix)
a_matrix %*% subB
```

```
##         b   c
## one   365 395
## two   414 450
## three 463 505
## four  512 560
```

4. Create three vectors `x,y,z` with integers and each vector has 3 elements. Combine the three vectors to become a 3×3 matrix `A` where each column represents a vector. Change the row names to `a,b,c`.

```r
x <- c(1,2,3)
y <- c(4,5,6)
z <- c(7,8,9)
A <- cbind(x,y,z)

rownames(A) <- c("a","b","c")
colnames(A) <- c("X","Y","Z")  #add column namess
print(A)
```

```
##   X Y Z
## a 1 4 7
## b 2 5 8
## c 3 6 9
```

```r
#if combined by rows:
B <- rbind(x,y,z)
print(B)
```

```
##   [,1] [,2] [,3]
## x    1    2    3
## y    4    5    6
## z    7    8    9
```