Class 7, Homework Assignment

Question 1 (2 points). Use a double loop to create a 9 rows by 9 columns matrix that contains the distance among all 9 locations. For example, the cell in the 2nd row and 3rd column should contain the distance between the second and third locations, given by

$$\sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2}$$

The geographic coordinates of each of the 9 locations is given below:

```
x <- rep(c(1,3,5), times=3)
y <- rep(c(1,3,5), each=3)
```

Solution:

```
dist <- matrix(, nrow=length(x), ncol=length(y))
for (i in 1:length(x)) {
    for (j in 1:length(y)) {
        dist[i,j] <- sqrt((x[i]-x[j])^2 + (y[i]-y[j])^2)
    }
}
dist
```

```
##           [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
##  [1,] 0.000000 2.000000 4.000000 2.000000 2.828427 4.472136 4.000000
##  [2,] 2.000000 0.000000 2.000000 2.828427 2.000000 2.828427 4.472136
##  [3,] 4.000000 2.000000 0.000000 4.472136 2.828427 2.000000 5.656854
##  [4,] 2.000000 2.828427 4.472136 0.000000 2.000000 4.000000 2.000000
##  [5,] 2.828427 2.000000 2.828427 2.000000 0.000000 2.000000 2.828427
##  [6,] 4.472136 2.828427 2.000000 4.000000 2.000000 0.000000 4.472136
##  [7,] 4.000000 4.472136 5.656854 2.000000 2.828427 4.472136 0.000000
##  [8,] 4.472136 4.000000 4.472136 2.828427 2.000000 2.828427 2.000000
##  [9,] 5.656854 4.472136 4.000000 4.472136 2.828427 2.000000 4.000000
##           [,8]     [,9]
##  [1,] 4.472136 5.656854
##  [2,] 4.000000 4.472136
##  [3,] 4.472136 4.000000
##  [4,] 2.828427 4.472136
##  [5,] 2.000000 2.828427
##  [6,] 2.828427 2.000000
##  [7,] 2.000000 4.000000
##  [8,] 0.000000 2.000000
##  [9,] 2.000000 0.000000
```

Question 2 (2 points). Use a double loop to create a 5 rows by 5 columns matrix. This matrix will contain the correlation between all the pairwise combinations of the 5 variables (columns) in the dataset below:

```
set.seed(1)
dat <- data.frame(matrix(rpois(20*5, lambda=3), nrow=20, ncol=5))
```

For example, the cell in the 2nd row and 3rd column should contain the correlation between the second and third variables. You can calculate this correlation using the code below:

```
cor(dat[,2], dat[,3])
```

```
## [1] 0.3245722
```

Solution:

```
dat.cor <- matrix(, nrow=dim(dat)[2], ncol=dim(dat)[2])
for (i in 1:dim(dat)[2]) {
    for (j in 1:dim(dat)[2]) {
        dat.cor[i,j] <- cor(dat[,i], dat[,j])
    }
}
dat.cor
```

```
##             [,1]         [,2]         [,3]         [,4]         [,5]
## [1,]  1.00000000 -0.391861658 -0.424607516 -0.317080551 -0.040093326
## [2,] -0.39186166  1.000000000  0.324572156  0.214862896  0.005581804
## [3,] -0.42460752  0.324572156  1.000000000  0.009575135 -0.574534729
## [4,] -0.31708055  0.214862896  0.009575135  1.000000000 -0.048177039
## [5,] -0.04009333  0.005581804 -0.574534729 -0.048177039  1.000000000
```

Question 3 (2 points). Create a matrix with integers from 1 to 16, 4 rows and 4 columns. Use a loop to calculate the minimum number for odd columns and maximum number for even columns and save these numbers in a vector.

Solution:

```
mat <- matrix(1:16, nrow=4, ncol=4)
mm <- numeric(dim(mat)[2])
for (i in 1:dim(mat)[2]) {
    mm[i] <- ifelse(i %% 2 == 1, min(mat[,i]), max(mat[,i]))
}
mat
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    5    9   13
## [2,]    2    6   10   14
## [3,]    3    7   11   15
## [4,]    4    8   12   16
```

```
mm
```

```
## [1]  1  8  9 16
```

Question 4 (2 points). Use the code provided below to create a lower triangular matrix. Use a double loop to fill in the upper triangular part of the matrix with the values in the lower triangular part to make it a symmetric matrix.

```
a <- matrix(rnorm(n=25, mean=0, sd=1), 5, 5)
a[upper.tri(a, diag = FALSE)] <- 0
a
```

```
##              [,1]       [,2]        [,3]      [,4]      [,5]
## [1,]  0.3981059  0.0000000  0.00000000 0.0000000  0.000000
## [2,] -0.6120264 -0.3672215  0.00000000 0.0000000  0.000000
## [3,]  0.3411197 -1.0441346  0.68973936 0.0000000  0.000000
## [4,] -1.1293631  0.5697196  0.02800216 0.1532533  0.000000
## [5,]  1.4330237 -0.1350546 -0.74327321 2.1726117 -1.253633
```

Solution:

```
for (i in 1:(dim(a)[1]-1)) {
    for (j in (i+1):dim(a)[1]) {
        a[i,j] <- a[j,i]
    }
}
a
```

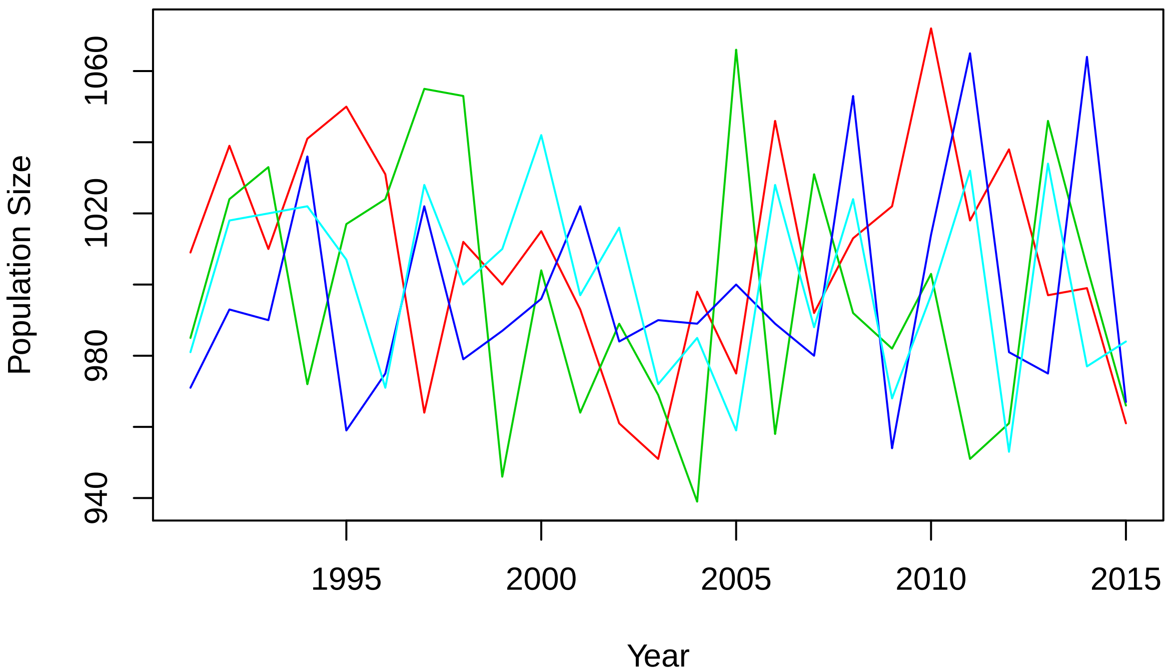
```
##              [,1]       [,2]        [,3]        [,4]       [,5]
## [1,]  0.3981059 -0.6120264  0.34111969 -1.12936310  1.4330237
## [2,] -0.6120264 -0.3672215 -1.04413463  0.56971963 -0.1350546
## [3,]  0.3411197 -1.0441346  0.68973936  0.02800216 -0.7432732
## [4,] -1.1293631  0.5697196  0.02800216  0.15325334  2.1726117
## [5,]  1.4330237 -0.1350546 -0.74327321  2.17261167 -1.2536334
```

Question 5 (2 points). Use the code provided below to create a data frame with 100 random numbers, 25 rows and 4 columns. Assuming that this data frame contains populatin size of four lizard species in 25 years (1991-2015). Plot a time series for each species. Only plot lines. Use a loop to plot all 4 species in one panel and use different colors to distinguish the species.

```
b <- data.frame(matrix(rpois(n=100, lambda=1000), nrow=25, ncol=4, byrow=T))
```

Solution:

```
years <- 1991:2015
plot(b[,1] ~ years, type='n', ylim=range(b), xlab='Year', ylab='Population Size')
for (i in 1:dim(b)[2]) {
    lines(b[,i] ~ years, col=i+1)
}
```


Population Size
940
980
1020
1060
1995
2000
2005
2010
2015
Year