Khan Inan
Transcriptomics 7633
Homework #1

1. What is an R object? (1pt)

An R object is essentially an individual instance of a way to represent data in R, whether it be a vector, matrix, dataframe or a list.

2. How many ways can data be assigned to an R object? (1pt)

There are many ways, it depends on the specific object, but there are many functions that can allow you to enter data into a variety of R objects

3. Why do you think it is important to ensure that data objects are of the correct type? (1pt)

It is important to ensure that they are the correct type because in the future if you decide to calculate or extract anything based on the data, you need to be sure that they are the right numeric or alpha type

4. What is the relationship between vectors, matrices, and data frames? (1pt)

A vector is any set of data, a matrix is a set of integers or numbers in X by X column/row format, and a data frame is any set of alpha or numeric values in X by X column/row format

5. Why might a data frame be more suitable than a matrix for holding heterogeneous biological data? (1pt)

A data frame would be more suitable mainly because it can hold words and qualitative data in addition to numerical values, which is present in heterogeneous biological data

1. Create a matrix (call it transcriptome) with the values below. The experiments are column names and genes are the row names. (3pts)

```
> transcriptome.data <- c(89, 78, 77, 56, 90, 99, 85, 97, 78, 94, 99, 87, 81, 83, 80, 79, 62, 51, 99, 88)
> transcriptome <- matrix(transcriptome.data,nrow=5,ncol=4,byrow=TRUE)
> rownames(transcriptome) <- c("GeneA", "GeneB", "GeneC", "GeneD", "GeneE")
> colnames(transcriptome) <- c("Control", "Nitrogen", "Phosphate", "Potassium")
> transcriptome
      Control Nitrogen Phosphate Potassium
GeneA      89       78        77        56
GeneB      90       99        85        97
GeneC      78       94        99        87
GeneD      81       83        80        79
GeneE      62       51        99        88
```

2. Use R code to calculate the average expression for each gene across all experiments. Call the vector (call it expression_average). (The vector should contain 5 values – one for each gene) (3pts)

```
> expression_average <- rowMeans(transcriptome)
> expression_average
GeneA GeneB GeneC GeneD GeneE
75.00 92.75 89.50 80.75 75.00
```

3. Sort the matrix so that the gene with the highest average expression value is on top and save it into a new data frame (call it “sorted_genes”) (4pts)

```
> sorted_genes <- transcriptome[order(-expression_average),]
> sorted_genes
      Control Nitrogen Phosphate Potassium
GeneB      90       99        85        97
GeneC      78       94        99        87
GeneD      81       83        80        79
GeneA      89       78        77        56
GeneE      62       51        99        88
```

a. Load the file expvalues.txt into R. (2pts)

```
> expvalues <- read.table("C://Users//khani//Downloads//expvalues.txt", header=TRUE)
```

b. The first three columns are “control” and the last three columns are “treatment” groups. Calculate the mean of the control samples and the mean of the treatment samples for each gene.(You can use a loop or apply functions) (5pts)

```
> ControlGroup <-expvalues [, c("Control1", "Control2", "Control3")]
> TreatmentGroup <- expvalues [, c("Treatment1", "Treatment2", "Treatment3")]
> Control_average <- rowMeans(ControlGroup)
> Treatment_average <- rowMeans(TreatmentGroup)
```

c. Calculate the fold change for each gene (fold change is the ratio of average treatment to average control). (2pts)

```
> foldChange <- Control_average/Treatment_average
> foldChange
  244901_at   244902_at   244903_at   244904_at   244905_at   244906_at   244907_at   244908_at   244909_at 244910_s_at
 1.29901227  0.59620276  1.07858523  1.56215373  0.26850170  1.31317346  1.31942038  0.78544757  0.82540674  1.95558797
  244911_at   244912_at   244913_at   244914_at 244915_s_at   244916_at   244917_at   244918_at   244919_at 244920_s_at
 1.72297232  1.02127017  3.84310949  0.43274568  1.46635867  0.47643064  0.49922984  1.92148366  1.25124527  1.01987356
244921_s_at 244922_s_at 244923_s_at   244924_at   244925_at 244926_s_at   244927_at 244928_s_at   244929_at   244930_at
 0.65105334  1.08577843  0.83955705  0.64453232  2.53927688  1.17979042  0.79186466  1.13782627  1.02040018  2.19794334
  244931_at   244932_at   244933_at   244934_at   244935_at   244936_at   244937_at   244938_at   244939_at   244940_at
 1.59545745  0.75613057  0.80537645  1.01871666  0.86470995  0.94557127  0.65082110  0.96488175  0.73108614  0.92740651
  244941_at   244942_at   244943_at 244944_s_at   244945_at   244946_at   244947_at   244948_at   244949_at   244950_at
 1.72369787 19.64522050  2.39678874  1.01968975  0.49755773  0.81500478  1.93819681  2.02568681  1.04399958  0.84188984
244951_s_at   244952_at 244953_s_at 244954_s_at   244955_at 244956_s_at   244957_at   244958_at 244959_s_at   244960_at
 1.08946142  2.05473562  1.00145392  1.56160789  0.87060118  0.28429524  0.70953642  0.45197915  0.99061984  1.14911006
  244961_at   244962_at   244963_at   244964_at   244965_at   244966_at   244967_at   244968_at   244969_at   244970_at
 0.93432153  0.91795324  1.23707016  0.88475468  0.78425908  1.15244371  1.04869296  1.03094751  1.31791161  0.85393122
```

d. Take the log2 of the fold change. You have just calculated log fold change(LFC). (2pts)

```
> log2(foldChange)
    244901_at     244902_at     244903_at     244904_at     244905_at     244906_at     244907_at     244908_at
 0.3774150599 -0.7461250397  0.1091401801  0.6435364343 -1.8969968649  0.3930574953  0.3999042945 -0.3484131160
    244909_at     244910_s_at   244911_at     244912_at     244913_at     244914_at     244915_s_at   244916_at
-0.2768228690  0.9676024357  0.7848995285  0.0303645676  1.9422740797 -1.2084086632  0.5522380239 -1.0696619091
    244917_at     244918_at     244919_at     244920_s_at   244921_s_at   244922_s_at   244923_s_at   244924_at
-1.0022239253  0.9422207099  0.3233646121  0.0283902972 -0.6191523566  0.1187297227 -0.2522997379 -0.6336753957
    244925_at     244926_s_at   244927_at     244928_s_at   244929_at     244930_at     244931_at     244932_at
 1.3444177149  0.2385305934 -0.3366742112  0.1862803013  0.0291350617  1.1361541978  0.6739701366 -0.4032927136
    244933_at     244934_at     244935_at     244936_at     244937_at     244938_at     244939_at     244940_at
-0.3122648107  0.0267528500 -0.2097118005 -0.0807418883 -0.6196670786 -0.0515759559 -0.4518866886 -0.1087262416
    244941_at     244942_at     244943_at     244944_s_at   244945_at     244946_at     244947_at     244948_at
 0.7855069243  4.2961064555  1.2611027528  0.0281302596 -1.0070641568 -0.2951195797  0.9547150731  1.0184111347
```

e. How many genes have a LFC > 1 OR < -1 ? (2pts)

```
> length(logfoldChange[logfoldChange > 1])
[1] 2394
> length(logfoldChange[logfoldChange < 1])
[1] 20416
>
```

f. Save the names of the genes that have a LFC > 1 into a file called “Induced_genes.txt” (2pts)

```
> write(logfoldChange, "C://Users//khani//Downloads//Induced_genes.txt")
>
```

g. Using the same set of induced genes in the previous question, create a boxplot to show the distribution of values for each induced gene in each experiment. The x-axis should have all six experiments, and the y-axis is the expression level. Save the boxplot as a pdf file called “boxplot.pdf” (5pts)

```
> boxplot <- boxplot(expvalues)
```