# HW 3 - Due Tuesday Sept 20, 2016. Upload R file to Moodle with name: HW3_490IDS_YOURNETID.R
# Do Not remove any of the comments. These are marked by #
# The .R file will contain your code and answers to questions.

#Name: Zixin Ouyang

# Main topic: Using the "apply" family function

#Q1 (5 pts)
# Given a function below,

```
myfunc <- function(z) return(c(z,z^2, z^3%/%2))
```

#(1) Examine the following code, and briefly explain what it is doing.

```
y = 2:8
myfunc(y)
matrix(myfunc(y),ncol=3)
```

### Your explanation

The code concatenates the number 2-8, the squares of 2-8, and the interger division of the cube of 2-8, and makes these numbers a matrix.

#(2) Simplify the code in (1) using one of the "apply" functions and save the result as m.
###code & result

```
m<-t(sapply(2:8,"myfunc",simplify = TRUE))
m
     [,1] [,2] [,3]
[1,]    2    4    4
[2,]    3    9   13
[3,]    4   16   32
[4,]    5   25   62
[5,]    6   36  108
[6,]    7   49  171
[7,]    8   64  256
```

#(3) Find the row product of m.
###code & result

```
apply(m,1,prod)
32    351   2048   7750  23328  58653 131072
```

#(4) Find the column sum of m in two ways.
###code & result

```
apply(m,2,sum)
colSums(m)
35 203 646
```

#(5) Could you divide all the values by 2 in two ways?
### code & result

```
m/2
apply(m,1:2,function(x) x/2)
     [,1] [,2]  [,3]
[1,]  1.0  2.0   2.0
```

```
[2,]  1.5  4.5   6.5
[3,]  2.0  8.0  16.0
[4,]  2.5 12.5  31.0
[5,]  3.0 18.0  54.0
[6,]  3.5 24.5  85.5
[7,]  4.0 32.0 128.0
```

```
#Q2 (8 pts)
#Create a list with 2 elements as follows:
l <- list(a = 1:10, b = 11:20)
#(1) What is the product of the values in each element?
lapply(l,prod)
$a
[1] 3628800
$b
[1] 670442572800
#(2) What is the (sample) variance of the values in each element?
lapply(l,var)
$a
[1] 9.166667
$b
[1] 9.166667
#(3) What type of object is returned if you use lapply? sapply? Show your R code that finds these answers.
class(lapply(l,var))
[1] "list"
class(sapply(l,var))
[1] "numeric"
```

```
# Now create the following list:
l.2 <- list(c = c(21:30), d = c(31:40))
#(4) What is the sum of the corresponding elements of l and l.2, using one function call?
mapply(sum,l,l.2)
a   b
310 510
#(5) Take the log of each element in the list l:
lapply(l,log)
a
[1] 0.0000000 0.6931472 1.0986123 1.3862944 1.6094379 1.7917595 1.9459101 2.0794415
2.1972246 2.3025851
$b
[1] 2.397895 2.484907 2.564949 2.639057 2.708050 2.772589 2.833213 2.890372 2.944439
2.995732
#(6) First change l and l.2 into matrixes, make each element in the list as column,
### your code here
l<-sapply(l,unlist)
```

```
l.2<-sapply(l.2,unlist)
#Then, form a list named mylist using l,l.2 and m (from Q1) (in this order).
### your code here
mylist<-list(l,l.2,m)
#Then, select the first column of each elements in mylist in one function call (hint '[' is the select
operator).
### your code here
lapply(mylist,function(x) x[,1])


#Q3 (3 pts)
# Let's load our friend family data again.
load(url("http://courseweb.lis.illinois.edu/~jguo24/family.rda"))
#(1) Find the mean bmi by gender in one function call.
tapply(fbmi,fgender,mean)
#(2) Could you get a vector of what the type of variables the dataset is made of？
class(objects(family))
"character"
#(3) Could you sort the firstName in height descending order?
fnames[order(fheight,decreasing=TRUE)]
"Joe" "Tom" "Tom" "Liz" "Jon" "Tim" "Bob" "Ann" "Dan" "Art" "Sal" "May"
"Sue" "Zoe"

#Q4 (2 pts)
# There is a famous dataset in R called "iris." It should already be loaded
# in R for you. If you type in ?iris you can see some documentation. Familiarize
# yourself with this dataset.
#(1) Find the mean petal length by species.
### code & result
tapply(iris$Petal.Length,iris$Species,mean)
setosa versicolor virginica
1.462      4.260      5.552
#(2) Now obtain the sum of the first 4 variables, by species, but using only one function call.
### code & result
by(iris[,1:4],iris$Species,colSums)
iris$Species: setosa
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
   250.3             171.4         73.1             12.3
------------------------------------------------------------------------
  iris$Species: versicolor
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
     296.8          138.5         213.0         66.3
------------------------------------------------------------------------
  iris$Species: virginica
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
       329.4         148.7          277.6         101.3
```

#Q5 (2 pts)
#Below are two statements, their results have different structure,

```
lapply(1:4, function(x) x^3)
sapply(1:4, function(x) x^3)
```

# Could you change one of them to make the two statements return the same results (type of object)?

```
as.list(sapply(1:4, function(x) x^3))
as.numeric(lapply(1:4, function(x) x^3))
```

#Q6. (5 pts) Using the family data, fit a linear regression model to predict
# weight from height. Place your code and output (the model) below.

```
lmfit<-lm(fweight~fheight)
Call:
lm(formula = fweight ~ fheight)
Coefficients:
(Intercept)     fheight
-455.666        9.154
summary(lmfit)
Call:
lm(formula = fweight ~ fheight)

Residuals:
  Min      1Q   Median     3Q     Max
-27.554  -9.689  -0.055  11.944  23.214

Coefficients:
               Estimate   Std. Error  t value    Pr(>|t|)
(Intercept) -455.666      107.029     -4.257     0.00111 **
fheight        9.154        1.594      5.741     9.29e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 16.94 on 12 degrees of freedom
Multiple R-squared:  0.7331,         Adjusted R-squared:  0.7109
F-statistic: 32.96 on 1 and 12 DF,  p-value: 9.287e-05
```

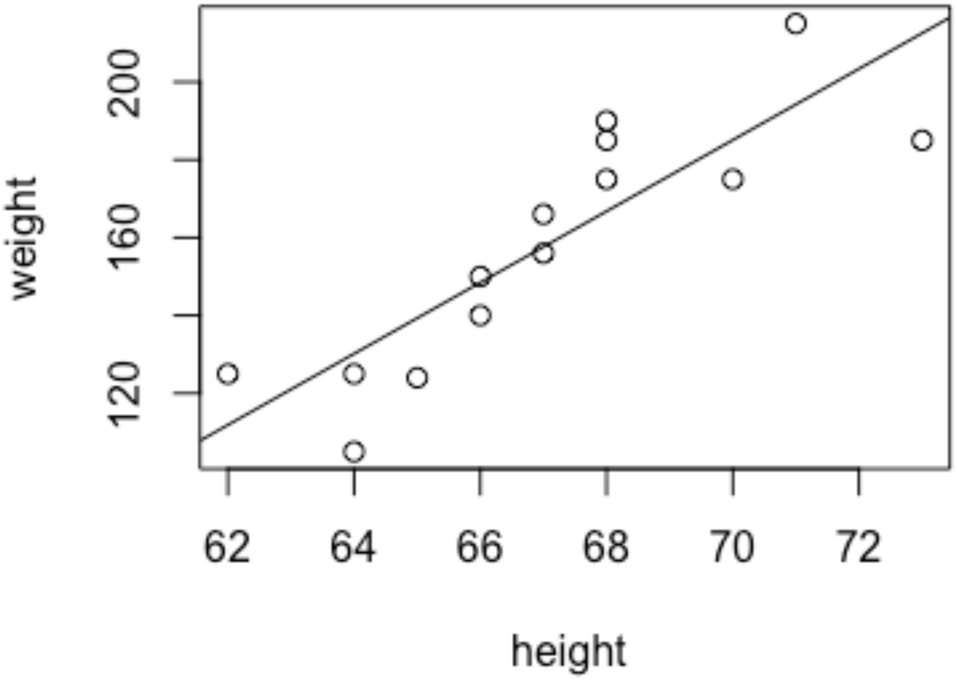
# How do you interpret this model?

In the first "Coefficients" table, the intercept is -455.666 and the coefficient for the height is 9.154. Therefore, the complete regression equation is weight = -455.666 + 9.154*height. The predicted weight for the family will increase by 9.154 for every one percent increase in the height.

In the second "Coefficients" table, the entries in the first column represent the least squares estimates, and the values in the second column correspond to the standard errors of each estimate.

# Create a scatterplot of height vs weight. Add the linear regression line you found above.
# Provide an interpretation for your plot.

```
plot(fheight, fweight, xlab="height ", ylab="weight")
abline(lmfit)
```



The linear model has a slope larger than zero. The higher the height, the heavier the weight. There are some outliers that do not fit the linear model.