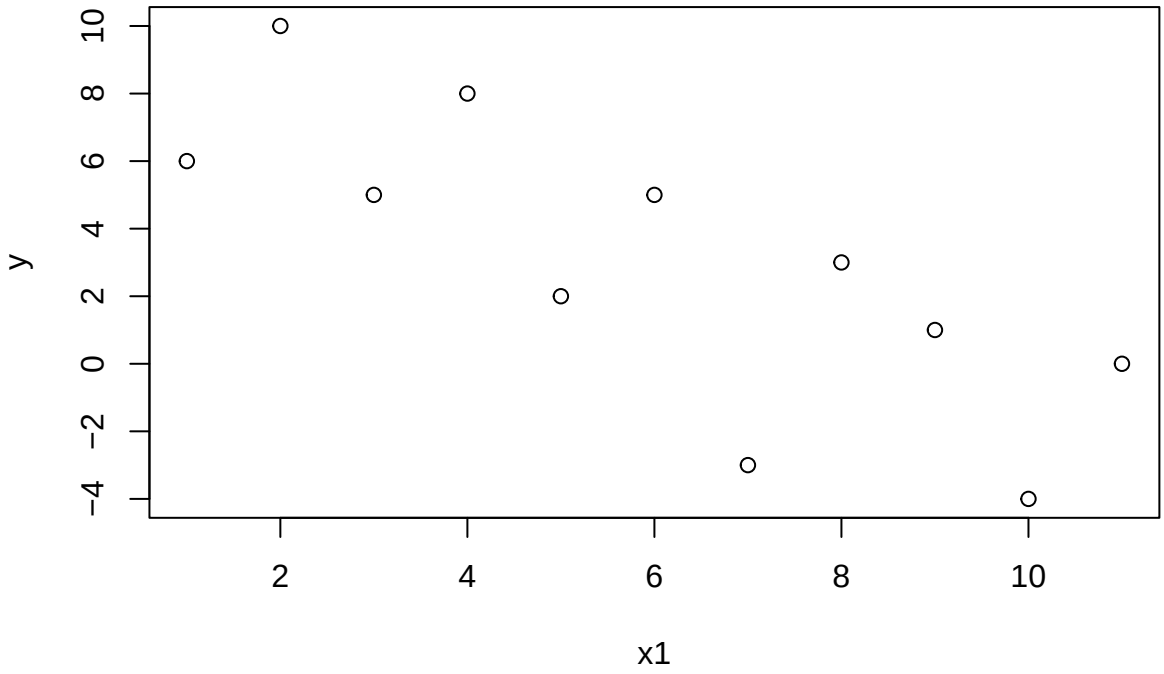
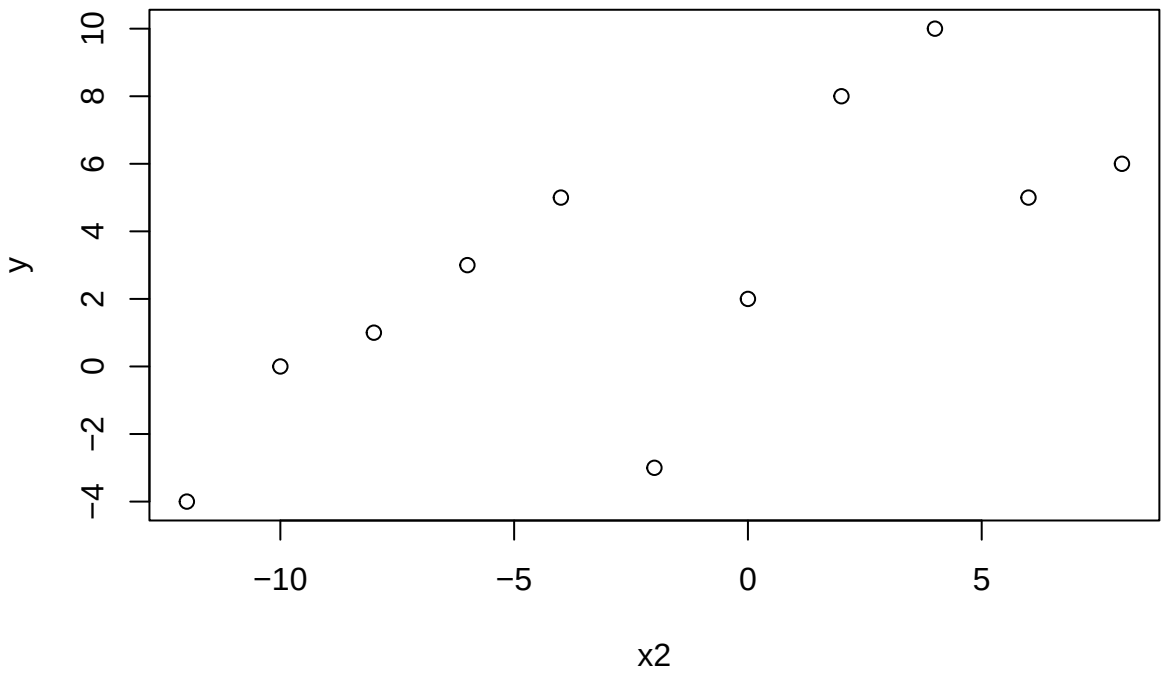
Assignment 2

Q1)

```
#1.1
q5a <- data.frame(
x1 <- c(1, 4, 9 , 11 , 3, 8 , 5, 10 , 2 , 7 , 6),
x2 <- c(8, 2, -8, -10, 6, -6, 0, -12, 4 , -2, -4),
y <- c(6, 8, 1 , 0 , 5, 3 , 2, -4 , 10, -3, 5)
)

ones <- rep(1,length(x1))

X <- matrix(c(ones,x1,x2),ncol = 3)

plot(x1,y)
```




```
plot(x2,y)
```




```
#1.2
Beta <- solve(t(X)%*%X)%*%(t(X)%*%y)
Beta
```

```
##      [,1]
## [1,] 14.0
## [2,] -2.0
## [3,] -0.5
```

Strictly judging based on the beta estimate. I believe Beta 1 has the largest relative importance

```
#1.3
Yhat<- X%*%Beta #Regression with x1 and x2
errors <- y - Yhat #Errors of model with x1 and x2
VarErrors <- sum(errors^2)/(length(y)-length(Beta)) #Variance of Errors for Regression Model (MSErrors)

varcovar <- solve(t(X)%*%X)*c(VarErrors)
varcovar
```

```
##           [,1]       [,2]       [,3]
## [1,] 37.149071 -7.2204301 -3.4731183
## [2,] -7.220430  1.4362519  0.6985407
## [3,] -3.473118  0.6985407  0.3590630
```

Here we see that Var(B2) is smaller than Var(B1) which indicates that there is less variability around the line when looking strictly at the plot between x2 and y. This means that Beta 2 is a better estimator than Beta 1.

```
#1.4

Sdotdot <- function(x, y) sum( (x - mean(x)) * (y - mean(y)) )

#Building Simple Linear Regression for y~x1:

Sxx_x1 <- Sdotdot(x1, x1)
Sxy_x1 <- Sdotdot(x1, y)
```

```
Beta1_x1 <- Sxy_x1 / Sxx_x1
Beta0_x1 <- mean(y) - Beta1_x1 * mean(x1)
Yhat_x1 <- Beta0_x1 + Beta1_x1*x1

#Building Simple Linear Regression for y~x2:

Sxx_x2 <- Sdotdot(x2, x2)
Sxy_x2 <- Sdotdot(x2, y)
Beta1_x2 <- Sxy_x2 / Sxx_x2
Beta0_x2 <- mean(y) - Beta1_x2 * mean(x2)
Yhat_x2 <- Beta0_x2 + Beta1_x2*x2

errors_x1<- y - Yhat_x1 #Errors of model with parameter x1
errors_x2<- y - Yhat_x2 #Errors of model with parameter x2

SSRx2_x1 <- sum(errors_x1^2) - sum(errors^2) #SSR(x2|x1)
SSRx1_x2 <- sum(errors_x2^2) - sum(errors^2) #SSR(x1|x2)

SSRx2_x1
```

```
## [1] 5.918182
```

```
SSRx1_x2
```

```
## [1] 23.67273
```

As we can see from comparing the extra sum of squares, since S(x2|x1) < S(x1|x2), x1 contributes more to the model when x2 is already in the model.

```
#1.5
anova(lm(y~x1+x2))
```

```
## Analysis of Variance Table
##
## Response: y
##           Df  Sum Sq Mean Sq F value   Pr(>F)
## x1         1 116.082 116.082 13.6567 0.006082 **
## x2         1   5.918   5.918  0.6963 0.428256
## Residuals  8  68.000   8.500
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(lm(y~x2+x1))
```

```
## Analysis of Variance Table
##
## Response: y
##           Df Sum Sq Mean Sq F value   Pr(>F)
## x2         1 98.327  98.327  11.568 0.009344 **
## x1         1 23.673  23.673   2.785 0.133702
## Residuals  8 68.000   8.500
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As we can see the anova table gives us the same result as our manually calculated Extra Sum of Squares. The value 116.082 corresponds to the sum of squares regression for a model only containing x1, while 5.918 is the extra sum of squares given that x2 is already in the model. Similarly, the 98.327 is the sum of squares

regression when only x2 is in the model, where as 23.673 is the extra sum of squares when x1 is already in the model. We can then conclude that x1 contributes more positively to a good fir out our model.

1.6)

We need to choose labels such that x1 explains y and such that x1 implies x2 or such that the correlation between y and x1 is much largr then that of y and x2 (to a certain extent)

An example of axis labels could be:

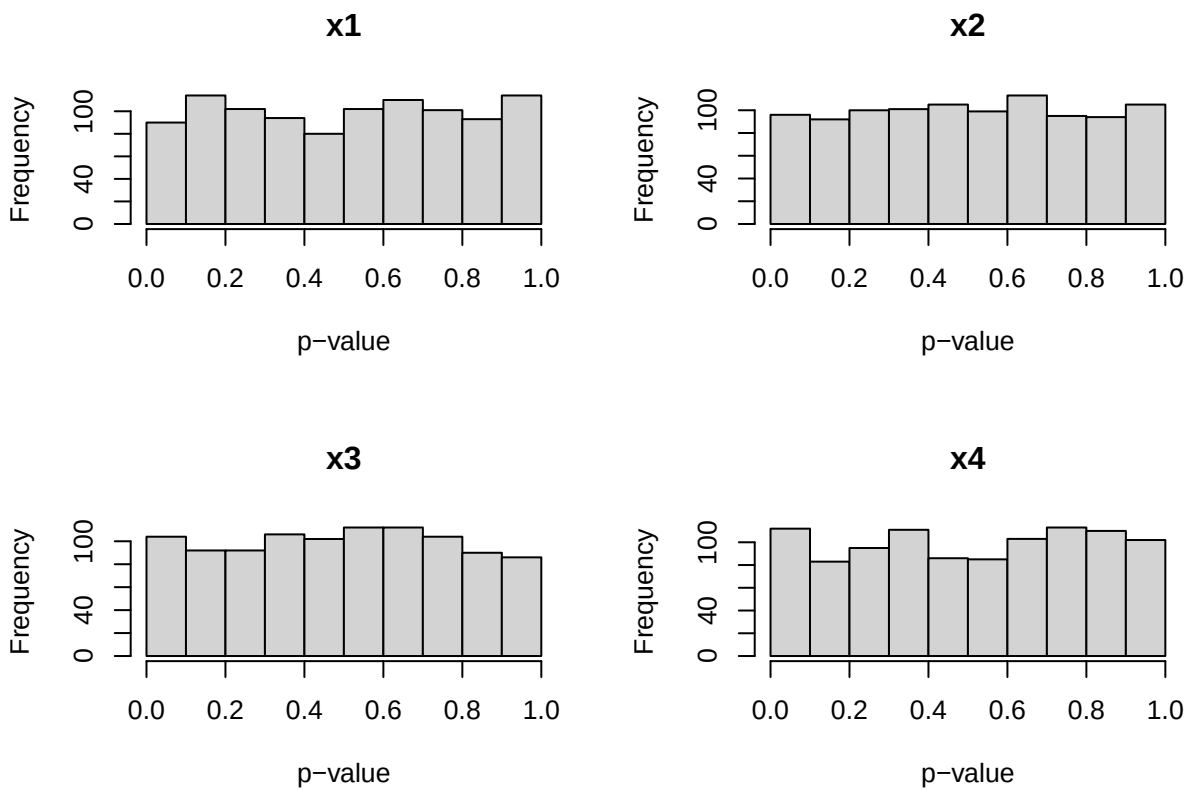
x1 - Wingspan x2 - Average Finger length y - Body Mass of a man

Question 2

2.1) To calculate sum of squares for multiple linear regression, R simply calculates the residuals by subtracting the true value from the predicted value of y to find the residuals. It then takes the sum of the squares of each difference to find the sum of squares residuals.

```
#2.2
n <- 100
x1 <- runif(n, 0, 10); x2 <- runif(n, 0, 10)
x3 <- runif(n, 0, 10); x4 <- runif(n, 0, 10)
sigma <- 2
reps <- 1000

# Prepare empty vectors :
p_values_x1 <- c()
p_values_x2 <- c()
p_values_x3 <- c()
p_values_x4 <- c()

for (i in 1:reps) {
y <- 0*x1 + 0*x2 + 0*x3 + 0*x4 + rnorm(n, 0, sigma)
mylm <- lm(y ~ x1 + x2 + x3 + x4)

# Record the p-values from anova():

anova_table <- anova(mylm)
p_values <- anova_table$"Pr(>F)"

p_values_x1[i] <- p_values[1]
p_values_x2[i] <- p_values[2]
p_values_x3[i] <- p_values[3]
p_values_x4[i] <- p_values[4]
}
```

```
#2.3

# Plot the results:

par(mfrow = c(2, 2))
hist(p_values_x1, main = "x1", xlab = "p-value")
hist(p_values_x2, main = "x2", xlab = "p-value")
hist(p_values_x3, main = "x3", xlab = "p-value")
hist(p_values_x4, main = "x4", xlab = "p-value")
```

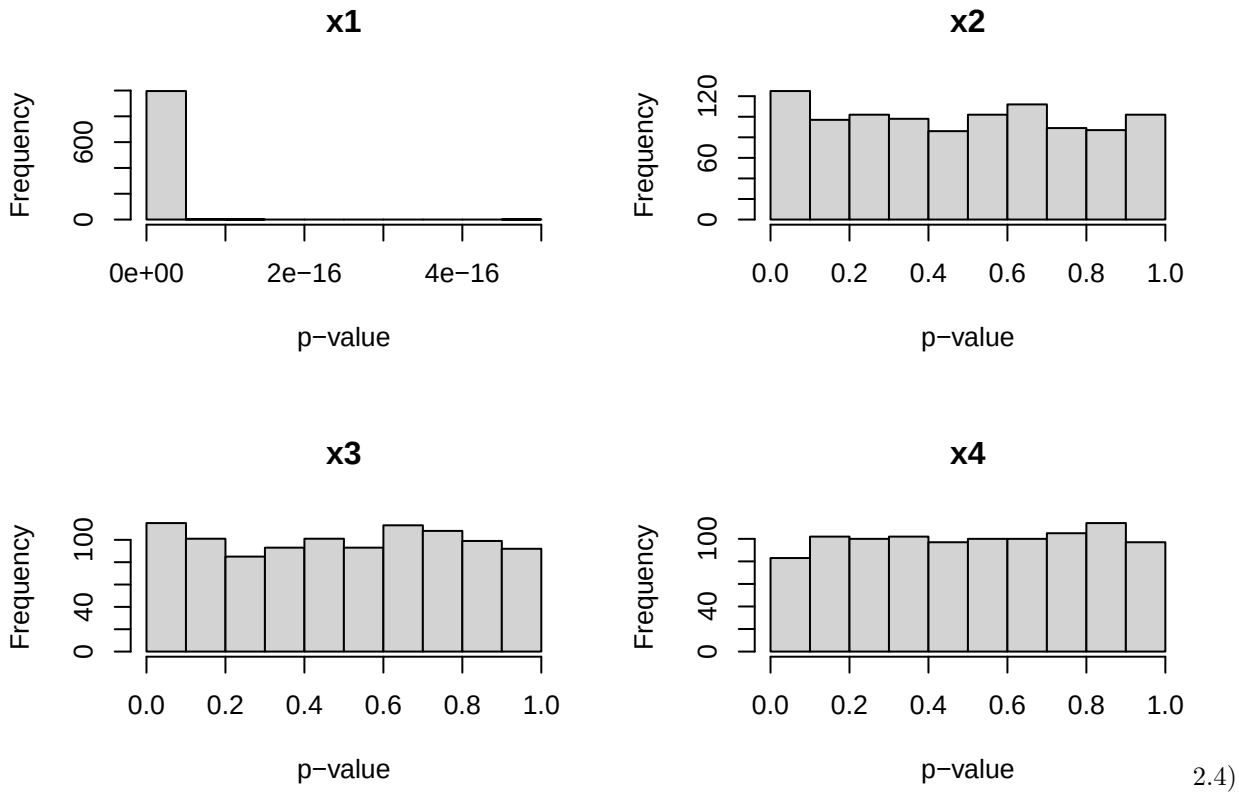


2.3)

Yes they appear to follow a uniform ditribution

```
#2.4
n <- 100
x1 <- runif(n, 0, 10); x2 <- runif(n, 0, 10)
x3 <- runif(n, 0, 10); x4 <- runif(n, 0, 10)
sigma <- 2
reps <- 1000

# Prepare empty vectors :
p_values_x1 <- c()
p_values_x2 <- c()
p_values_x3 <- c()
p_values_x4 <- c()

for (i in 1:reps) {
y <- 1*x1 + 0*x2 + 0*x3 + 0*x4 + rnorm(n, 0, sigma)
mylm <- lm(y ~ x1 + x2 + x3 + x4)

# Record the p-values from anova():

anova_table <- anova(mylm)
p_values <- anova_table$"Pr(>F)"

p_values_x1[i] <- p_values[1]
p_values_x2[i] <- p_values[2]
p_values_x3[i] <- p_values[3]
p_values_x4[i] <- p_values[4]
```

```
}

# Plot the results:

par(mfrow = c(2, 2))
hist(p_values_x1, main = "x1", xlab = "p-value")
hist(p_values_x2, main = "x2", xlab = "p-value")
hist(p_values_x3, main = "x3", xlab = "p-value")
hist(p_values_x4, main = "x4", xlab = "p-value")
```

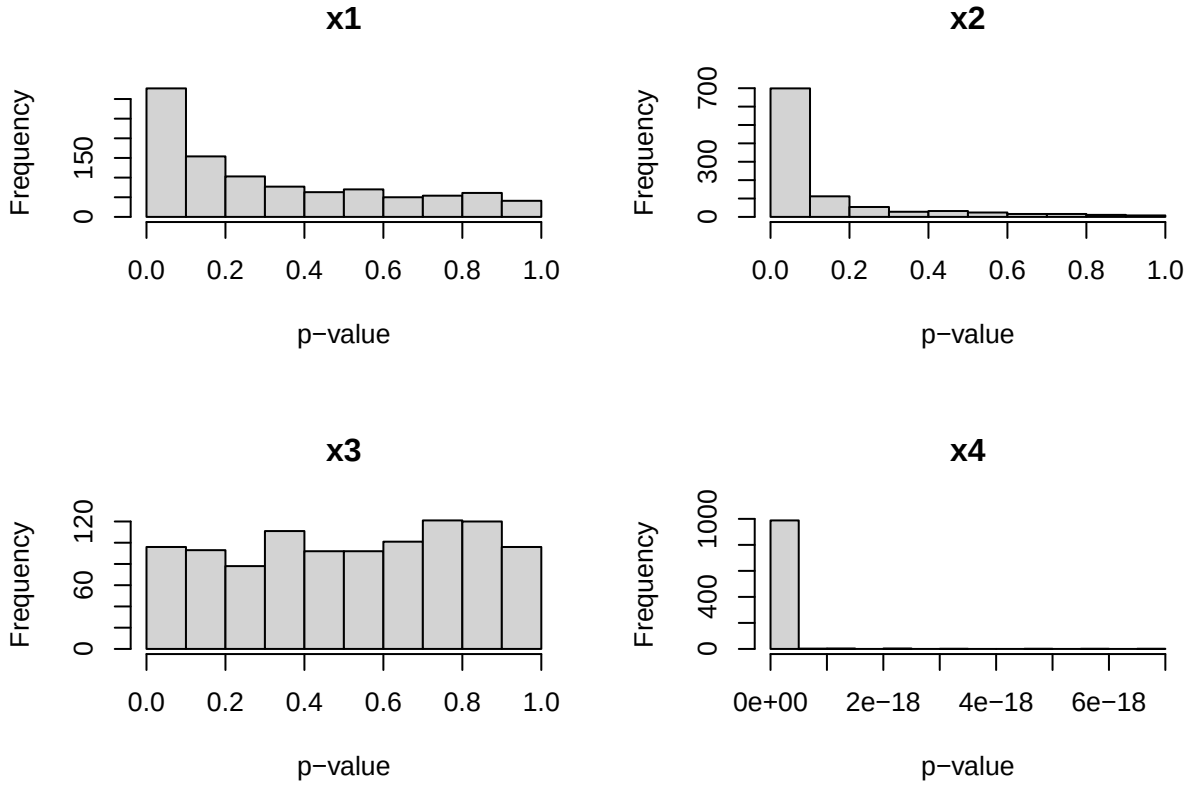


2.4)

The histograms for the p-values of x2, x3, x4 are still uniformly distributed however the p-values for x1 are all equal to 0. This means that there is a 0% chance that the null hypothesis be rejected (B1 = 0)

```
#2.5
n <- 100
x1 <- runif(n, 0, 10); x2 <- runif(n, 0, 10)
x3 <- runif(n, 0, 10); x4 <- runif(n, 0, 10)
sigma <- 2
reps <- 1000

# Prepare empty vectors :
p_values_x1 <- c()
p_values_x2 <- c()
p_values_x3 <- c()
p_values_x4 <- c()

for (i in 1:reps) {
y <- 0*x1 + 0*x2 + 0*x3 + 1*x4 + rnorm(n, 0, sigma)
mylm <- lm(y ~ x1 + x2 + x3 + x4)
```

```r
# Record the p-values from anova():

anova_table <- anova(mylm)
p_values <- anova_table$"Pr(>F)"

p_values_x1[i] <- p_values[1]
p_values_x2[i] <- p_values[2]
p_values_x3[i] <- p_values[3]
p_values_x4[i] <- p_values[4]
}

# Plot the results:

par(mfrow = c(2, 2))
hist(p_values_x1, main = "x1", xlab = "p-value")
hist(p_values_x2, main = "x2", xlab = "p-value")
hist(p_values_x3, main = "x3", xlab = "p-value")
hist(p_values_x4, main = "x4", xlab = "p-value")
```




```r
anova(mylm)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df Sum Sq Mean Sq  F value    Pr(>F)
## x1         1   0.48    0.48   0.1224  0.727199
## x2         1  38.99   38.99   9.8769  0.002233 **
## x3         1   2.19    2.19   0.5553  0.458010
```

```
## x4         1 908.62  908.62 230.1493 < 2.2e-16 ***
## Residuals 95 375.06    3.95
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

2.5 Written)

Setting B4 = 1 and everything else 0 made the p-values for x4 = 0. Which logically makes sense given that the test for B4 = 0 will always be false.

As for the skewness of the other predictor variables, theoretically speaking, changing B4 = 1 should not have any impact on the uniformity of the other predictor variables and thus they should be uniformly distributed. However, given the fact that the ANOVA table first calculates SSR for x1, then x2, then x3, then x4, it may be picking up some correlation (by chance) between x1, x2, x3 and x4 which can cause the significance test for B1 = B2 = B3 = 0 not necessarily always be true