Question 3 & Question 5

Question 3

1.)

```
# importing data
x<-c(4,3,5,7)
y<-c(6,2,4,11)


#plot to see how data looks
plot(x,y,pch=19)

#calculating mean of x and y
xmean<-mean(x)
ymean<-mean(y)

xmean
```

```
## [1] 4.75
```

```
ymean
```

```
## [1] 5.75
```

```
#summary of x and y
summary(x)
```

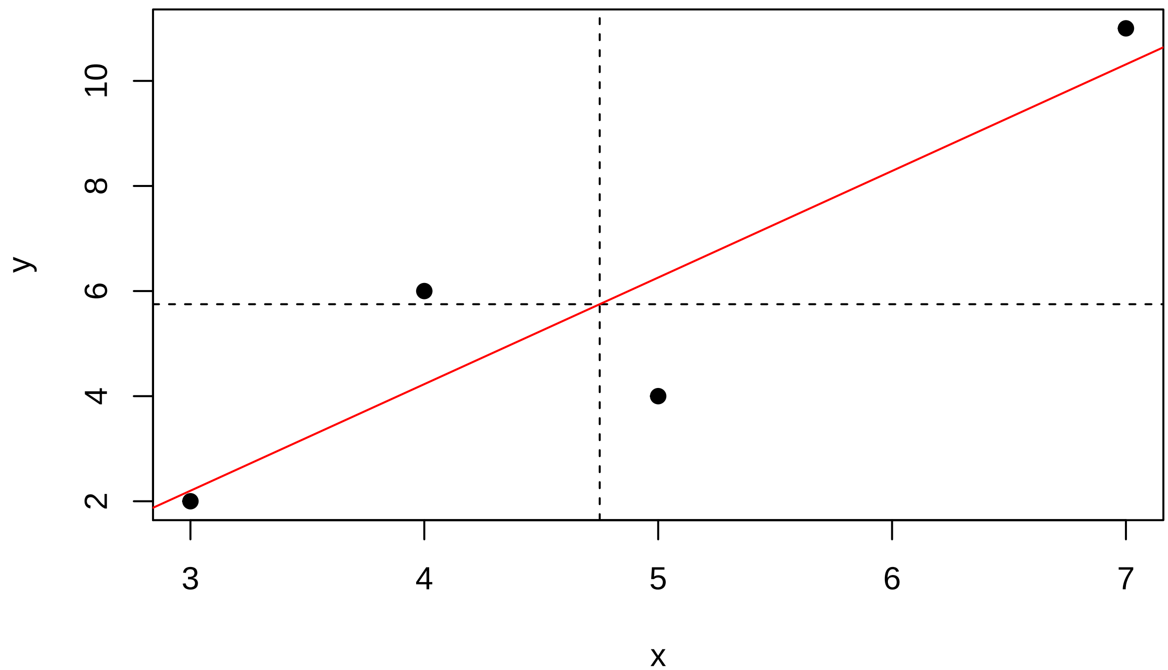
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3.00    3.75    4.50    4.75    5.50    7.00
```

```
summary (y)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.00    3.50    5.00    5.75    7.25   11.00
```

```
# putting mean of x & y on plot for  better visualization
abline(v=xmean,h=ymean,lty=2)

#fitting linear regression line on data
fit.RP<-lm(y~x)
abline(coef(fit.RP),col='red')
```




```r
#summary of linear regression
summary(fit.RP)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##       1       2       3       4
##  1.7714 -0.2000 -2.2571  0.6857
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -3.8857     3.5164  -1.105    0.384
## x             2.0286     0.7068   2.870    0.103
##
## Residual standard error: 2.091 on 2 degrees of freedom
## Multiple R-squared:  0.8046, Adjusted R-squared:  0.7069
## F-statistic: 8.237 on 1 and 2 DF,  p-value: 0.103
```

2.)

```r
#putting x and y into matrix form
x<-c(1,1,1,1,4,3,5,7)

X<-matrix(x,nrow=4,ncol=2)
Y<-matrix(y,nrow=4,ncol=1)

# Computing Xtranspose X
XtX<-t(X)%*%X

XtX
```

```
##      [,1] [,2]
## [1,]    4   19
## [2,]   19   99
```

```r
# Computing inverse Xtranspose X
XtXinv<-solve(XtX)

XtXinv
```

```
##            [,1]       [,2]
## [1,]  2.8285714 -0.5428571
## [2,] -0.5428571  0.1142857
```

```r
#Computing B hat using forumla B hat = (XtX)^-1* XtY

XtY<-t(X)%*%Y
Bhat<-XtXinv%*%XtY

Bhat
```

```
##           [,1]
## [1,] -3.885714
## [2,]  2.028571
```

3.)

```r
#Computing Hat Matrix with formula X*XtX^-1*Xt
Hat<-X%*%(XtXinv)%*%t(X)

Hat
```

```
##            [,1] [,2]      [,3]        [,4]
## [1,] 0.31428571  0.4 0.2285714  0.05714286
## [2,] 0.40000000  0.6 0.2000000 -0.20000000
## [3,] 0.22857143  0.2 0.2571429  0.31428571
## [4,] 0.05714286 -0.2 0.3142857  0.82857143
```

4.)

```r
x<-c(4,3,5,7)

##making functions to calculate Sxx,Sxy,B1 hat,SST,SSres,MSres

##*** These functions will help for future calculations in Q3 and Q5*****

SXX<-function(x)
{
    return (sum((x - mean(x))^2))
}

SXY<-function(x,y)
{
    return (sum((x - mean(x))*(y-mean(y))))
}

B1_hat<-function(x,y)
{
    return (SXY(x,y)/SXX(x))
}

SST<-function(x,y)
{
    return (sum(y^2)-((sum(y)^2)/length(y)))
}

SSres<-function(x,y)
{
    return (SST(x,y)-(B1_hat(x,y)*SXY(x,y)))
}

MSres<-function(x,y)
{
    return (SSres(x,y)/(length(x)-2))
}


## Calulating Sigma Squared
sigma_squared_hat<-MSres(x,y)
```

5.)

```r
## Calculating Sigma Squared * (XtX)^1
B_hat_Variance<-XtXinv*sigma_squared_hat

B_hat_Variance
```

```
##            [,1]       [,2]
```

```
## [1,] 12.364898 -2.3730612
## [2,] -2.373061  0.4995918
```

6.)

```r
## Functions for calculator Estimated Standard Error for B1 and B0

ESE_B1<-function(x,y)
{
    return (sqrt(MSres(x,y)/SXX(x)))
}


ESE_B0<-function(x,y)
{
  return (sqrt(MSres(x,y)*((1/length(x))+((mean(x)^2)/SXX(x)))))
}

## ESE B1 hate and B0 hat

ESE_B1(x,y)
```

```
## [1] 0.7068181
```

```r
ESE_B0(x,y)
```

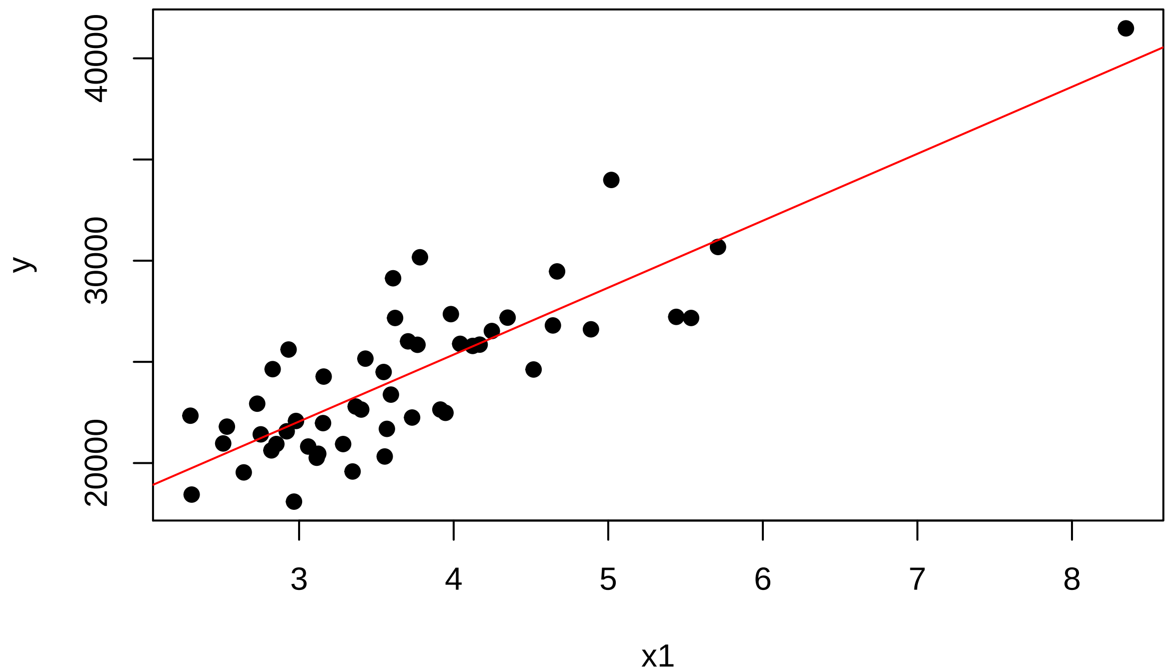
```
## [1] 3.516376
```

Question 5

```r
#important data for Q5

file1<-"http://www.math.mcgill.ca/yyang/regression/data/salary.csv"
salary <-read.csv(file1 ,header=TRUE)
x1<-salary$SPENDING/1000
y<-salary$SALARY

#plotting data with regression line for visualization
plot(x1,y,pch=19)
fit.Salary<-lm(y~x1)
abline(coef(fit.Salary),col='red')
```




```r
summary(fit.Salary)
```

```
##
## Call:
## lm(formula = y ~ x1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3848.0 -1844.6  -217.5  1660.0  5529.3
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  12129.4     1197.4   10.13 1.31e-13 ***
## x1            3307.6      311.7   10.61 2.71e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2325 on 49 degrees of freedom
## Multiple R-squared:  0.6968, Adjusted R-squared:  0.6906
## F-statistic: 112.6 on 1 and 49 DF,  p-value: 2.707e-14
```

1.)

```r
# function for B0 hat

B0_hat<-function(x,y)
{
    return (mean(y)-(B1_hat(x,y))*mean(x))
}

B1<-B1_hat(x1,y)
B0<-B0_hat(x1,y)




B1
```

```
## [1] 3307.585
```

```r
B0
```

```
## [1] 12129.37
```

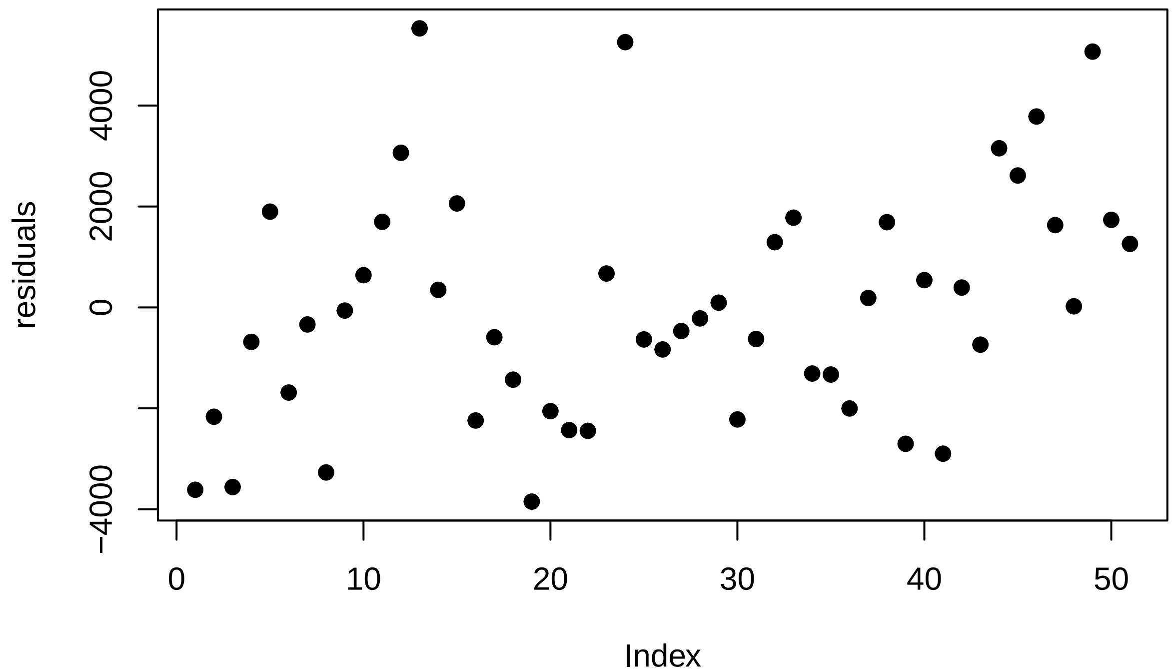
```
As we can see the ouputs of the R code match the entries in the Estimate column,
thus it produces the correct results
```

2.)

```r
#calculating y hat entries and residuals (e)
y_hat=B1*x1+B0

residuals=y-y_hat

#plotting residuals for visulalization
plot(residuals,pch=19)
```




```r
# summation of resisuals
round(sum(residuals))
```

```
## [1] 0
```

```r
# summation of resisuals * (x- mean of x)
round(sum(residuals*(x1-mean(x1))))
```

```
## [1] 0
```

```r
# summation of resisuals * (y hat)
round(sum(residuals*y_hat))
```

```
## [1] 0
```

```
As we can see all the summations equal to 0,
thus we have verified the orthogoanlity results
```

3.)

```
To calculate the Standard Error of Intercept Directly,
we use the formula t0=B0/ESE(B0), thus ESE(B)
is equal to 12129.4/10.13 which is 119.684
```

```r
##Calculating Std Error using data directly

ESE_B0(x1,y)
```

```
## [1] 1197.351
```

4.)

```r
#Calculating Residual standard error
residual_standard_error<-sqrt(MSres(x1,y))
residual_standard_error
```

```
## [1] 2324.779
```

5.)

```
(Null Hypothesis) H_0: B1=0

(Alternative Hypothesis) H_1: B1!=0

Failing to Reject H_0 proves no linear association between SALARY
and SPENDING

p-value: 1.31e-13, we reject the null if our p-value<=alpha

1.31e-13<=0.05 thus we reject the Null Hypothesis and prove their
is a linear association between SALARY and SPENDING
```

6.)

```r
#Functions for SSr and F-statistic

SSR<-function(y,y_hat)
{
    return (sum((y_hat-mean(y))^2))
}

F_statistic<-function(x,y,y_hat,p)
{
    return(((SSR(y,y_hat))/(p-1))/((SSres(x,y))/(length(x)-p)))
}


F_statistic(x1,y,y_hat,2)
```

```
## [1] 112.5995
```

7.)

```r
## Function for calculating SSt
SST_2<-function(x,y,y_hat)
{
    return (SSres(x,y)+SSR(y,y_hat))
}

SST_2(x1,y,y_hat)
```

```
## [1] 873380265
```

```r
SST(x1,y)
```

```
## [1] 873380265
```

```
We can see that these 2 formulas for SSt are equivalent
which verifies SSt=SSres+SSr for us
```

8.)

```r
#Functions for confidence interval
CI_B1<-function(x,y,alpha)
{
    B1<-B1_hat(x,y)

    ESE<-ESE_B1(x,y)


    interval=c(B1+(qt(p=(alpha/2),df=length(x)-2,lower.tail = T))*(ESE),B1+(qt(p=(alpha/2),df=length(x)

    return (interval)

}

CI_B1(x1,y,0.1)
```

```
## [1] 2784.997 3830.173
```

```
There is a 90% chance the true value of B1 lies within this interval
```

9.)

```r
#Function to generated y given x with the regression line

predicted_y<-function(x,y,input)
{
  return ((B1_hat(x,y)*(input/1000))+B0_hat(x,y))

}

predicted_y(x1,y,4800)
```

```
## [1] 28005.78
```

10.)

```r
# Making matrix form just like the question
x_new<-append(c(rep(1,length(x1))),x1)
X1<-matrix(x_new,nrow=length(x1),ncol=2)
Y<-matrix(y,nrow=length(y),ncol=1)


Bhat<-function(X,Y)
{
  return ((solve(t(X1)%*%X1))%*%(t(X1)%*%Y))
}

B<-Bhat(X1,Y)

Y_new<-function(B,x_new)
{
  X_new<-matrix(c(1,(x_new/1000)),nrow=1,ncol=2)
  return(X_new%*%B)
}

Y_predicted<-Y_new(B,4800)


# returning Y predicted using matrix format of oredicting
Y_predicted
```

```
##          [,1]
## [1,] 28005.78
```

```r
# the residual standard error, previously calculated
residual_standard_error
```

```
## [1] 2324.779
```