**Problem 1**

The R function to find the mean and standard deviation is

```
#Function for Problem 1
#Purpose is to find mean and standard deviation of all columns
#Mean is stored in column 1 of output
#Standard Deviation is stored in column 2 of output

Problem1Fun<-function(InTable){
        OutVec<-matrix(c(1:2*dim(InTable)[2]),ncol=2,nrow=dim(InTable)[2])
        for(i in 1:dim(InTable)[2]){
                OutVec[i,1]<-mean(InTable[,i])
                OutVec[i,2]<-sd(InTable[,i])
        }
        return(OutVec)
}
```

The output for running `Problem1Fun(Nitrates)` is

```
> Problem1Fun(Nitrates)
             [,1]         [,2]
 [1,]     2.00000 1.351725e+00
 [2,]    33.68870 8.366869e-02
 [3,]  -112.21931 3.152602e-01
 [4,]   483.08873 1.128625e+02
 [5,] 37431.11734 1.647951e+04
 [6,]   679.43042 1.093556e+03
 [7,] 26499.28471 1.576737e+04
 [8,]    26.52196 2.216925e+01
 [9,]    17.79275 7.931085e+01
[10,]     7.54878 5.698698e-01
[11,]   613.55278 1.320502e+03
```

To use the `apply` function to produce the means in 1 line, we would use the following command and the output is as follows

```
> apply(Nitrates,2,mean)
        LAND.USE.LEVEL1          latitude_decimals          longitude_decimal
                2.00000                   33.68870                 -112.21931
      elevation.Meters.  DistanceFromUrbanCenter                POP_DENSITY
              483.08873                37431.11734                  679.43042
      INCOME_PER_CAPITA                   VegCover                      NO3.N
            26499.28471                   26.52196                   17.79275
                     pH               conductivity
                7.54878                  613.55278
```

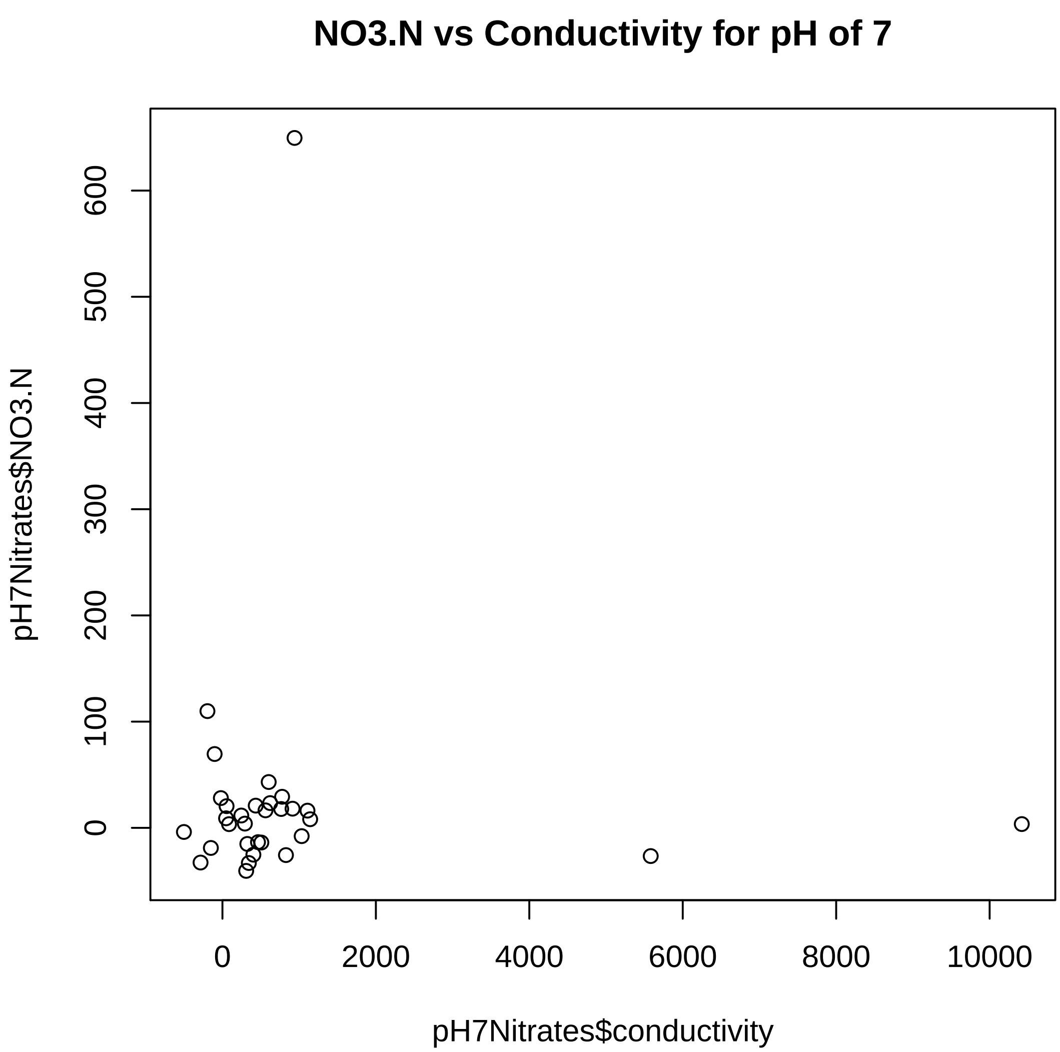
Similarly, we have the following for standard deviation:

```
> apply(Nitrates,2,sd)
        LAND.USE.LEVEL1          latitude_decimals          longitude_decimal
           1.351725e+00               8.366869e-02               3.152602e-01
      elevation.Meters.  DistanceFromUrbanCenter                POP_DENSITY
           1.128625e+02               1.647951e+04               1.093556e+03
      INCOME_PER_CAPITA                   VegCover                      NO3.N
           1.576737e+04               2.216925e+01               7.931085e+01
                     pH               conductivity
           5.698698e-01               1.320502e+03
```

**Problem 2**

We create a dataframe with rows with a pH value of 7, then plot NO3 against conductivity with the following code and then the output plot is the included

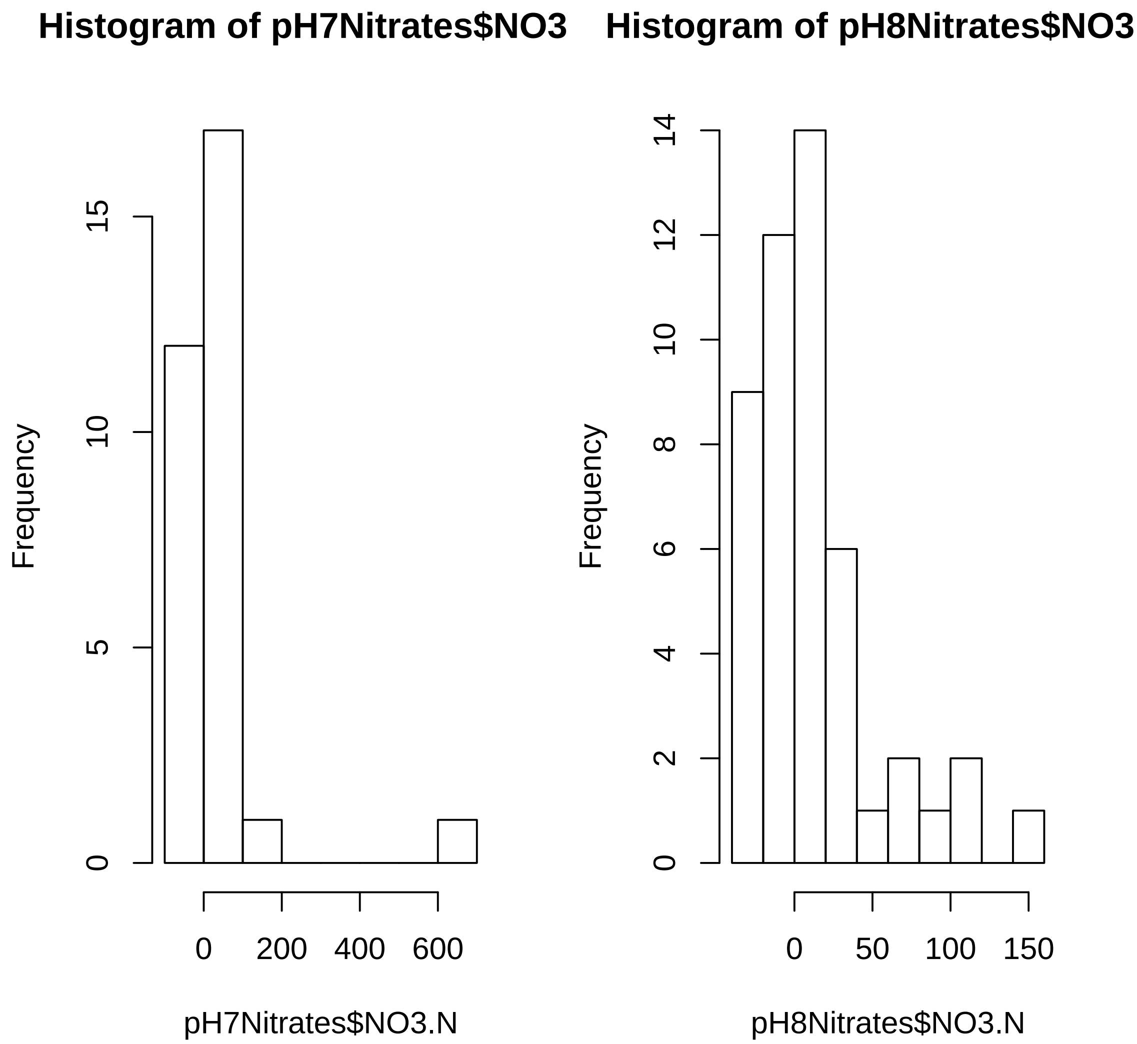
```
> pH7Nitrates<-Nitrates[which(Nitrates$pH==7),]
> plot(pH7Nitrates$conductivity,pH7Nitrates$NO3.N,
       main="NO3.N vs Conductivity for pH of 7")
> dev.print(device=pdf,"Problem2.pdf")
```




**Problem 3**

To produce a side-by-side histogram of NO3.N for pH of 7 and pH of 8, we create two dataframes and then use `hist()` which produces the subsequent plot.

```
> pH7Nitrates<-Nitrates[which(Nitrates$pH==7),]
> pH8Nitrates<-Nitrates[which(Nitrates$pH==8),]
> par(mfrow=c(1,2))
> hist(pH7Nitrates$NO3.N)
> hist(pH8Nitrates$NO3.N)
> dev.print(device=pdf,"Problem3.pdf")
```




**Problem 4**

To determine how many data points have a landuse of 2, we use the following line

```
> dim(Nitrates[which(Nitrates$LAND.USE.LEVEL1==2),])[1]
[1] 45
```

So, we see that there are 45 data points with a landuse of 2.

**Problem 5**

After installing packages with `install.packages('mvtnorm')` then using the following given code

```
library(mvtnorm)

for(i in 1:1000){
                currentrealization<-rmvnorm(1,c(0,0),CovMat)
                Realizations[i,1]<-currentrealization[1,1]
                Realizations[i,2]<-currentrealization[1,2]
}
```

we can create a covariance matrix with the given values and create a matrix of zeros

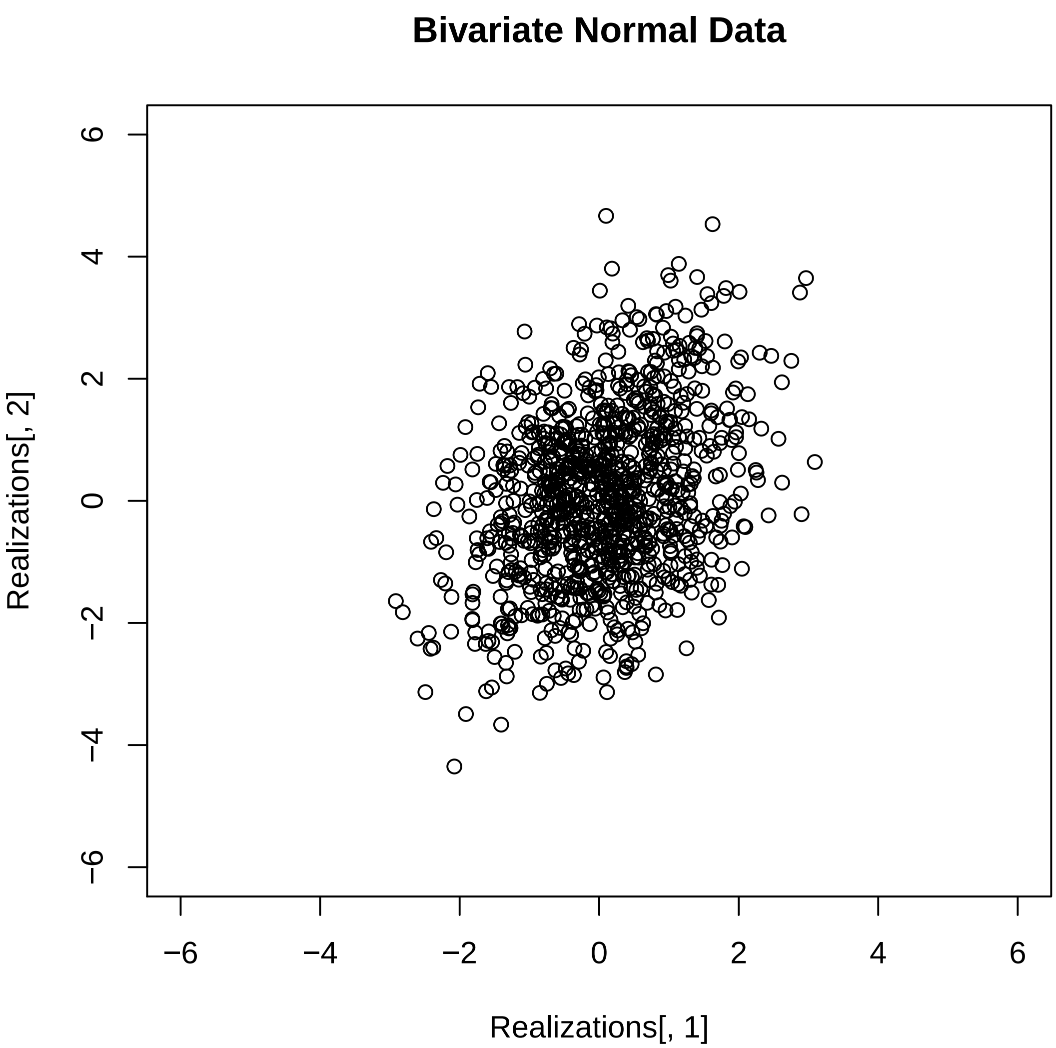
```
> CovMat<-matrix(rep(1,4),nrow=2,ncol=2)
> CovMat[1,]<-c(1,0.5)
> CovMat[2,]<-c(0.5,2)
> CovMat
     [,1] [,2]
[1,]  1.0  0.5
[2,]  0.5  2.0

> Realizations<-matrix(0,nrow=1000,ncol=2)
```

then using the given code, we have the following scatter-plot by

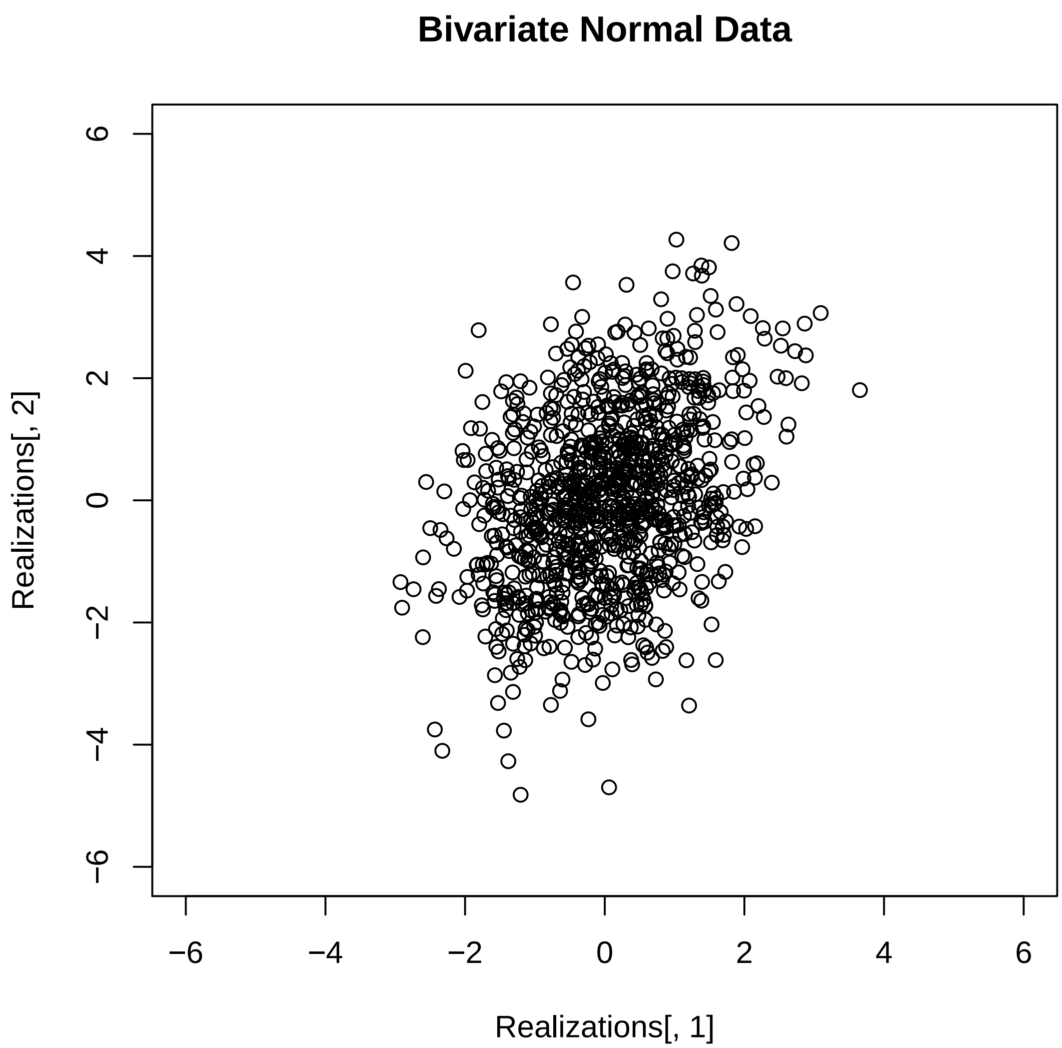
```
> plot(Realizations[,1],Realizations[,2],main="Bivariate Normal Data",
                       xlim=c(-6,6),ylim=c(-6,6))
> dev.print(device=pdf,"Problem5.pdf")
```

For this data we have the following as the means of the columns



```
> apply(Realizations,2,mean)
[1] -0.03239635 -0.03260434
```

We then run the problem again and we have the following scatter-plot and for this we have the following as




the means of the columns

```
> apply(Realizations,2,mean)
[1] 0.01286922 0.03089901
```

As we see for the means, both are close to 0, and as we ran the code for a second time, we see that (perhaps by co-incidence) we have means closer, in magnitude, to 0.

**Problem 6**

Using `rexp` we see the syntax of inputs is `rexp(n,1/mean)`, where n is the number of outcomes. So, for a mean of 1 and 50 outcomes, we can use `rexp(50,1)`. Thus, to produce the sum, we would use

```
> sum(rexp(50,1))
```

To produce this sum 1000 times, we use the following loop

```
realization<-rep(1,1000)
for( i in 1:1000){
        realization[i]<-sum(rexp(50,1))
}
```

Then using `length`, we have the percentage of sums above 60

```
> 100*length(which(realization >60))/length(realization)
[1]  8.8
```

Thus there are 8.8% of the 1000 sums above 60.

**Problem 7**
To find the number of iterations it takes to find the sum of 50 realizations over 80, we run

```
i<-0
testsum<80
while(testsum<80){
        testsum<-sum(rexp(50,1))
        i<-i+1
}
print(i)
```

Running this a few times, we require 1554 iterations, 16034 iterations, 3313 iterations to have the sum over 80.

**Problem 8**
Using the following code

```
> X<-rnorm(1000,0,1)
> Y<-rnorm(1000,0,2)
> Z<-X+Y
> hist(Z)
> dev.print(device=pdf,"Problem8.pdf")
```

we have the following histogram This plot seems to be a Normal distribution about 0.

**Problem 9** Using the following code

```
> X<-rnorm(1000,0,1)
> Y<-rnorm(1000,X,2)
> Z<-X+Y
> hist(Z)
> dev.print(device=pdf,"Problem9.pdf")
```

we have the following plot. Which still looks Normal about 0, but shows a bit of being skewed.

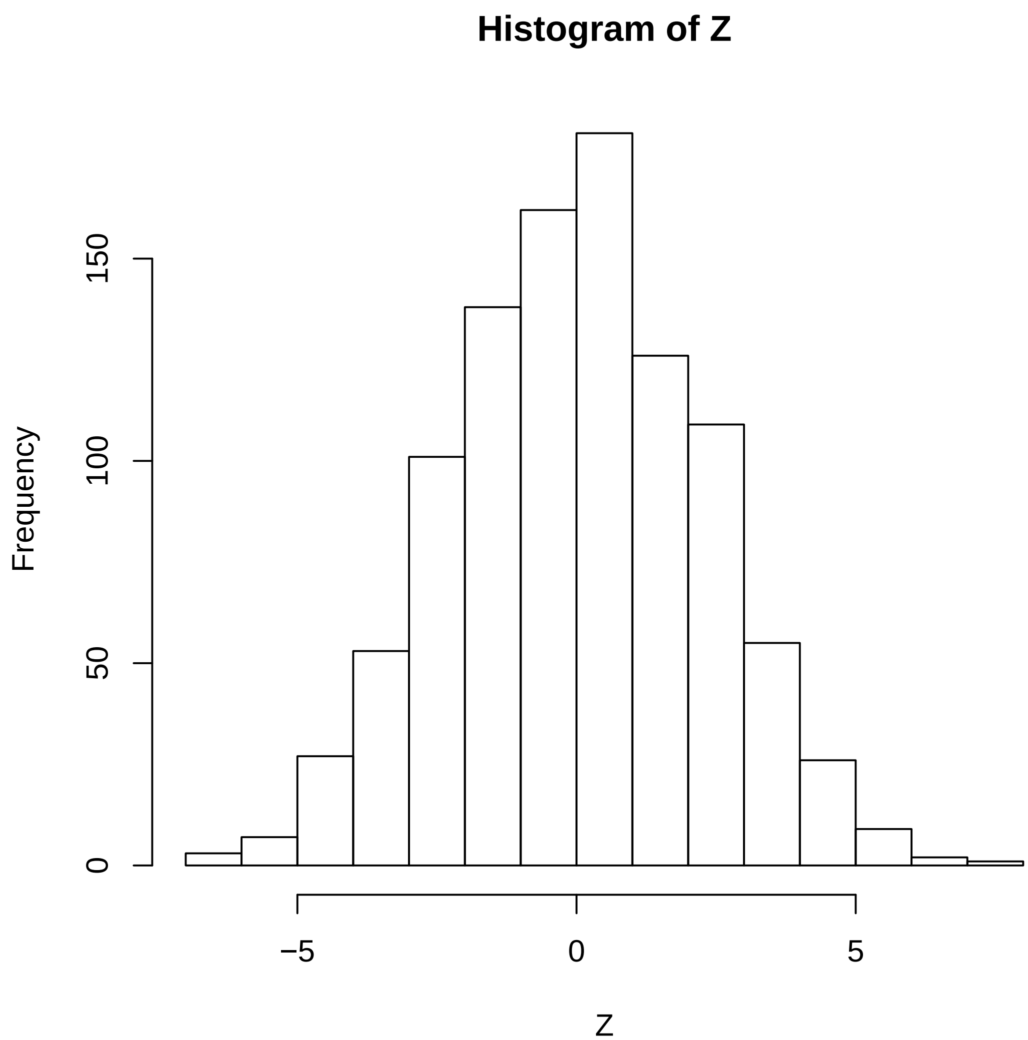



Figure 1: Problem 8 Plot

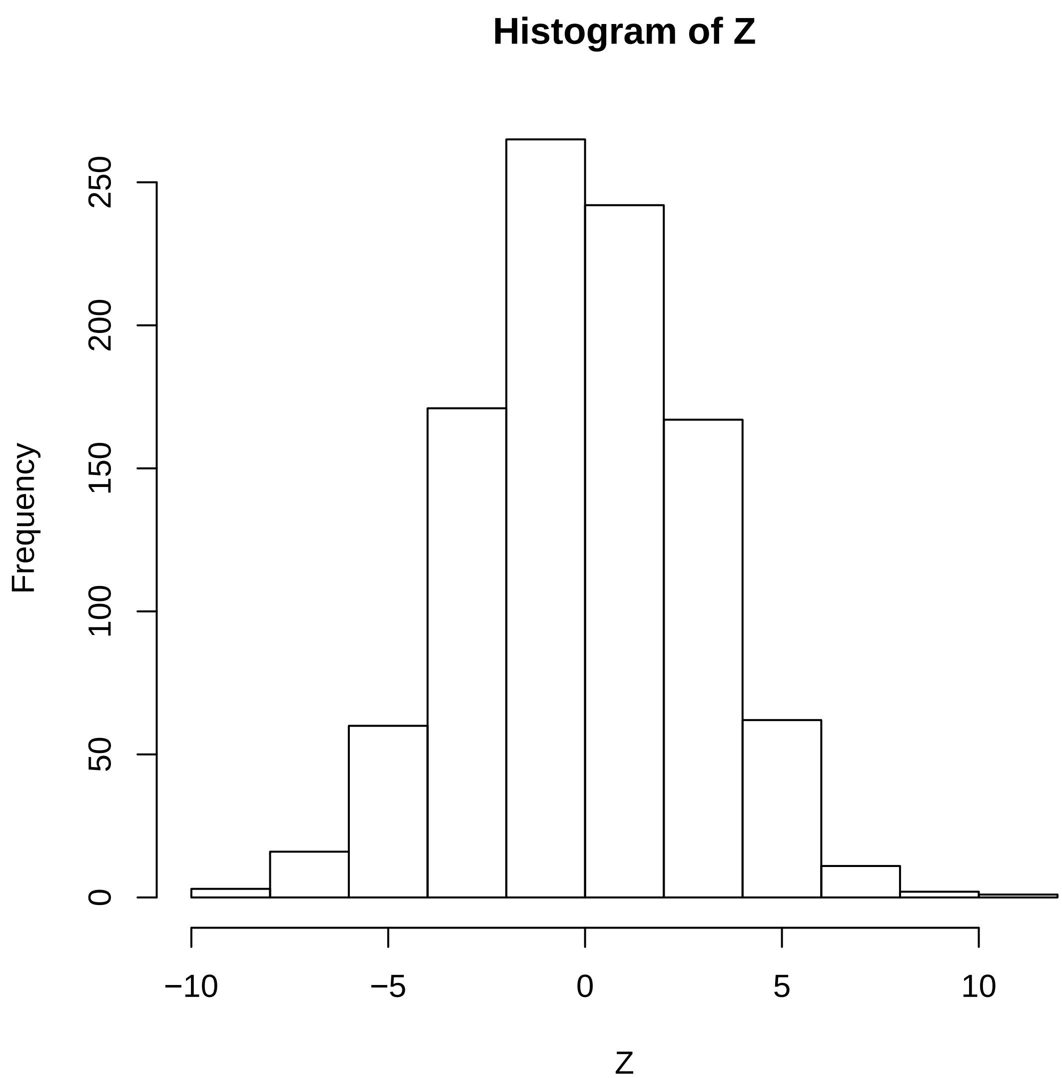



Figure 2: Problem 9 Plot