MINI PROJECT #5
Statistical Methods for Data Science
Group - 46

Anish Joshi(2021591978) and Aneena Manoj(2021623362)

Contribution – Firstly, both the teammates discussed the questions together and wrote the required R code for all the three questions. Further, the R code was executed by both of them and the findings were reported. Both the teammates worked efficiently to finish the project.

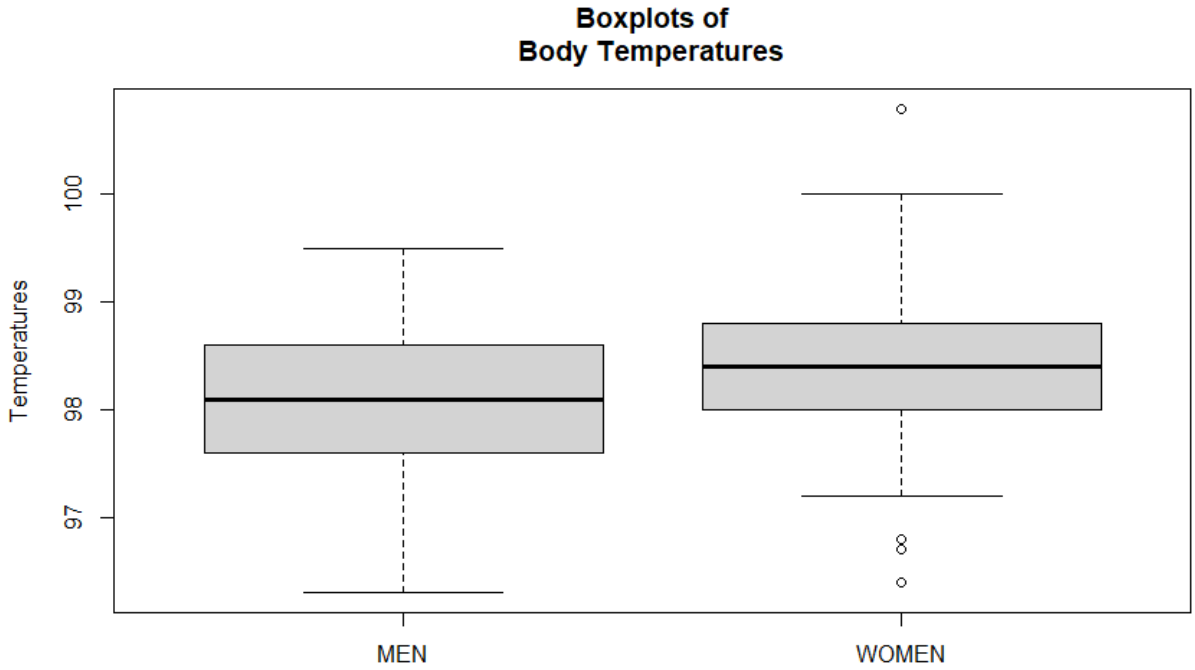
SECTION - 1

QUESTION – 1

1a) To determine the answer to this, we will gather summary statistics, and do a boxplot.
First, we read the csv using the read.csv function and separate it into two subsets for men and women using the subset function

```
> summary(men$body_temperature)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   96.3    97.6    98.1    98.1    98.6    99.5
> summary(women$body_temperature)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  96.40   98.00   98.40   98.39   98.80  100.80
```

We start by drawing boxplots of body temperature values for men and women



Observations:

- Q1, Median and Q3 are higher for females than males.
- The distribution of females can have a slightly higher mean value than of the males.
- There are more outliers in the women box plot implying more variability for them than the males. Hence we cannot assume equal variances.

Next we draw Q-Q plot of body temperature values for men and women

Observations:

- As we can see from the Q-Q plots, the distributions of the body temperature values for both men and women are approximately normal

Let 'M' be the body temperatures of males and 'F' be the body temperatures of females. So the sample mean m estimates the population mean m and the sample mean $\bar{f}$ estimates the population mean f

We take the null hypothesis : Difference between means is 0 => $m - \bar{f} = 0$
Alternate Hypothesis : Difference between means is not equal to 0 => $m - \bar{f} \neq 0$

The samples here are to be treated as independent samples, with unequal variances coming from an approximately normal distribution, hence we can use t distribution with Satterthwaite's approximation to get the confidence interval.

The confidence interval from the function t.test in R is (-0.53964856, -0.03881298) The p-value we got is 0.02394

```
> # confidence interval using t,test function for the body temperature values
> t.test(men$body_temperature, women$body_temperature, alternative =
+          'two.sided', var.equal = F)

        Welch Two Sample t-test

data:  men$body_temperature and women$body_temperature
t = -2.2854, df = 127.51, p-value = 0.02394
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.53964856 -0.03881298
sample estimates:
mean of x mean of y
 98.10462  98.39385
```

Since p-value is less than 0.05 and 0 does not lie in the confidence interval, we reject the null hypothesis and hence we know that the body temperature means of females and males are not equal. The width of the confidence interval is very small, hence the sample means differ by very small amounts. Mean of female body temperature is slightly higher than that of men.

1b)

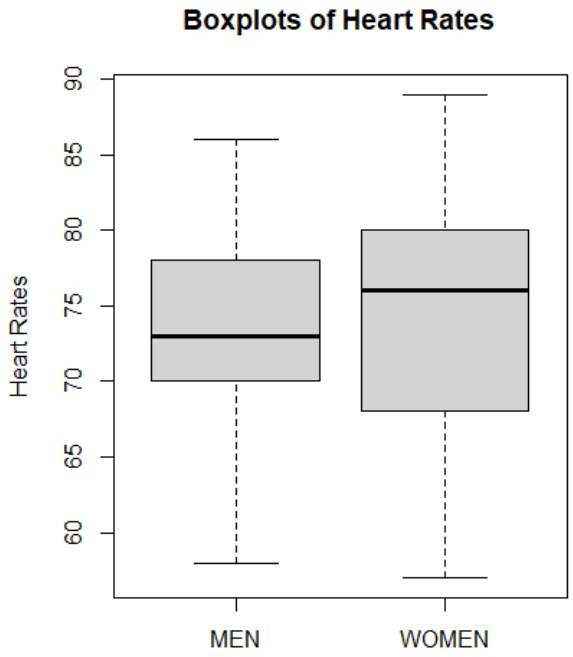
To determine the answer to this, we once again gather summary statistics, and do a boxplot.

```
> # Summary Statistics for heart rate values
> summary(men$heart_rate)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  58.00   70.00   73.00   73.37   78.00   86.00
> summary(women$heart_rate)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  57.00   68.00   76.00   74.15   80.00   89.00
```

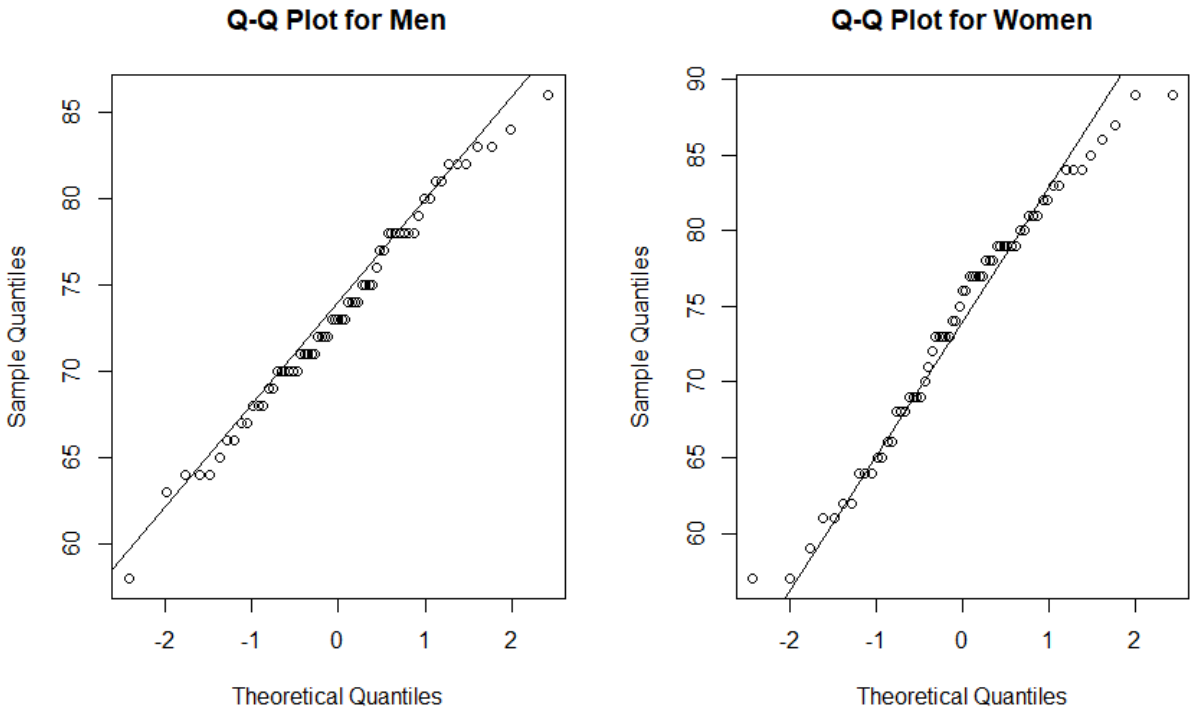
We then plot the boxplot of heart rate values for men and women:

Observations:

- Q1 for females is smaller than Q1 for men
- Median and Q3 are higher for females than for males.
- The values for females have more variability.



Moving onto Q-Q plots,



Observations:- From the Q-Q plots, the distributions of these heart rate values for both males and females are approximately normal.

Let ‘M’ be the body temperatures of males and ‘F’ be the body temperatures of females.

So the sample mean m estimates the population mean m and the sample mean $\bar{f}$ estimates the population mean f

We take the null hypothesis : Difference between means is 0 => $m - \bar{f} = 0$
Alternate Hypothesis : Difference between means is not equal to 0 => $m - \bar{f} \neq 0$

The samples here are to be treated as independent samples, with unequal variances coming from an approximately normal distribution, hence we can use t distribution with Satterthwaite’s approximation to get the confidence interval.

The confidence interval from the function t.test in R is (-3.243732, 1.674501) The p-value we got is 0.5287.

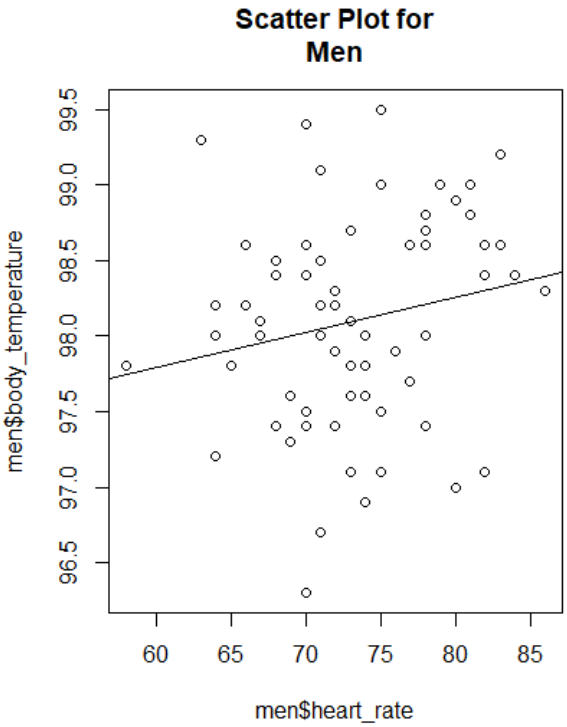
Since p-value is greater than 0.05 and the value 0 lies in the confidence interval, we accept the null hypothesis and hence the heart rate value means females and males are equal.

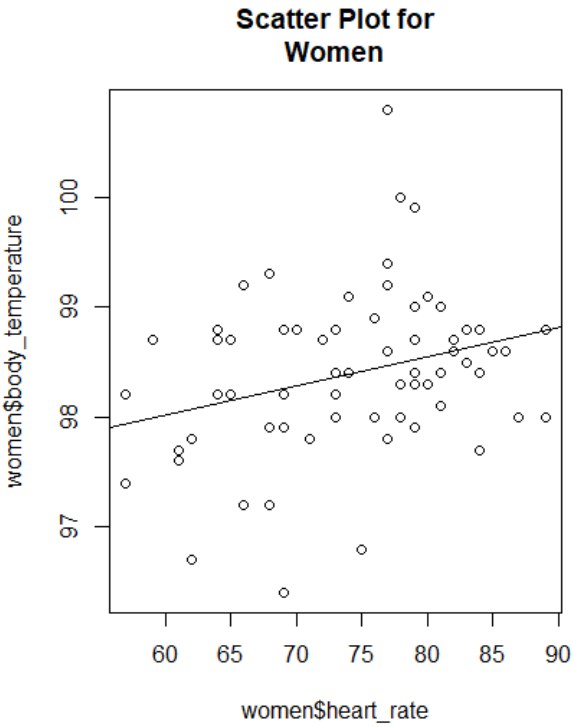
```
> # Confidence interval using the t.test function for the heart rate values
> t.test(men$heart_rate, women$heart_rate, alternative = 'two.sided', var.equal = F)

        Welch Two Sample t-test

data:  men$heart_rate and women$heart_rate
t = -0.63191, df = 116.7, p-value = 0.5287
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -3.243732  1.674501
sample estimates:
mean of x mean of y
 73.36923  74.15385
```

1c)
Next we draw and consider a scatter plot and further plot a regression line that reflects the linear relationship between them.

Observations:

- As we can see from the graph, the line drawn has a slope which is greater than 0.
- This suggests positive association of correlation between body temperature and heart rate values.
- Based on the graph, we can assume that the strength of their linear relationship is weak.





We get the correlation between two variables by using the function cor.

```
> # Finding the correlation values between body temperatures and heart rates
> cor(men$body_temperature, men$heart_rate)
[1] 0.1955894
> cor(women$body_temperature, women$heart_rate)
[1] 0.2869312
```

Correlation between body temperature and heart rate for males is: 0.1955894
Correlation between body temperature and heart rate for females is: 0.2869312

We know that the larger the value, the stronger the correlation, The relationship between the body temperature and heart rates is weak positive linear relationship. Since the correlation value for females is higher than males, we can say that for females the correlation between body temperature and heart rate is stronger than for males.

QUESTION – 2

2a)

To simulate Monte Carlo estimates of coverage probabilities and to construct the confidence intervals, we have created the following functions:

check_z_ci –

- takes n and λ values as input parameters
- Simulates a sample and constructs an interval and returns whether the true mean exists within the confidence Interval

```
> # Creating check_z_ci function - checks whether true mean exists within confidence interval
> # Simulating a sample and constructing an interval
> check_z_ci <- function(n, lambda) {
+
+   U <- rexp(n,lambda)
+   lbound <- mean(U) - qnorm(0.975) * sd(U) / sqrt(n)
+   ubound <- mean(U) + qnorm(0.975) * sd(U) / sqrt(n)
+   tm = 1/lambda
+   if(ubound>tm & lbound<tm) {
+     return (1)
+   }
+   else {
+     return (0)
+   }
+ }
```

zproportion –

- takes n and λ values as input parameters
- calls check_z_ci function 5000 times with replicate() and calculates the coverage probabilities

```
> # Creating function zproportion - compute coverage probabilities using checkztmci
> zproportion <- function(n, lambda) {
+   values <- replicate(5000, check_z_ci(n, lambda))
+   ones <- values[which (values == 1)]
+   return (length(ones)/5000)
+ }
>
> zproportion(5,0.01)
[1] 0.8068
```

mean_find –

- Generates samples from a distribution and returns the mean

```
> # Creating function mean_find - Returns mean of samples from distribution
> mean_find <- function(n,lambda) {
+   u.star <- rexp(n, lambda)
+   return (mean(u.star))
+ }
```

check_b_ci –

- using the n and λ given as input parameters
- Calls mean_find function 1000 times and forms the confidence interval and returns whether the true mean is present in the interval

```
> # Creating check_b_ci function - checks whether true mean exists within confidence interval
> # calls the mean_find func() 1000 times and forms the confidence interval
>
> check_b_ci <- function(n, lambda) {
+   U <- rexp(n,lambda)
+   tm <- 1/lambda
+   lambda1 = 1/mean(U)
+   V <- replicate(1000, mean_find(n,lambda1))
+   bound <- sort(V)[c(25, 975)]
+   if(bound[2]>tm & bound[1]<tm) {
+     return (1)
+   }
+   else {
+     return (0)
+   }
+ }
```

bproportion -

- takes n and λ as input parameters
- constructs a parametric initial bootstrap sample and calls check_b_ci 5000 times and calculates the coverage probabilities

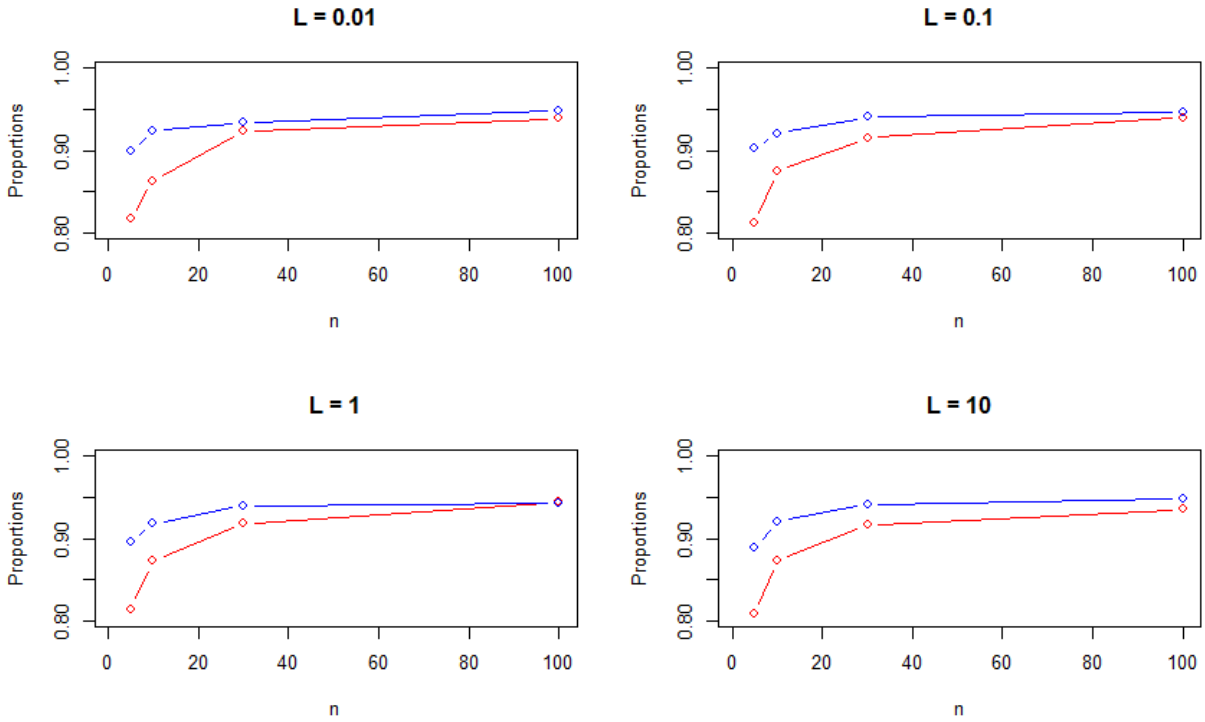
```
> # Creating function bproportion - constructs a parametric initial bootstrap sample
> # calls checkbtmci 5000 times and calculates coverage probabilities
>
> bproportion <- function(n, lambda) {
+   values <- replicate(5000, check_b_ci(n, lambda))
+   ones <- values[which (values == 1)]
+   return (length(ones)/5000)
+ }
>
> bproportion(5,0.01)
[1] 0.898
```

Using these functions, for the (n,λ) combination as (5, 0.01) we get the coverage probabilities as:
Z-interval: 0.8068
Bootstrap interval: 0.898

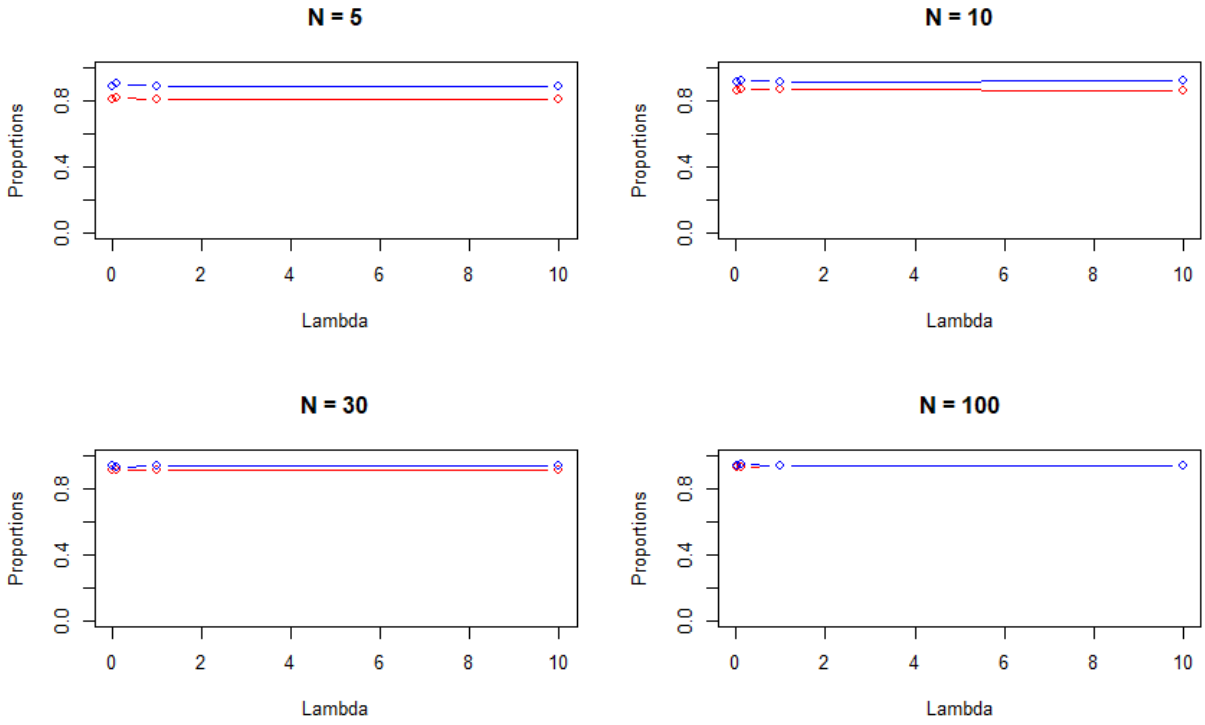
2b)

Repeat (a) for the remaining combinations of (n, λ). Present an appropriate summary of the results.

Z PROP	L = 0.01	L = 0.1	L = 1	L = 10
N = 5	0.8102	0.8162	0.8136	0.8086
N = 10	0.8628	0.8714	0.8760	0.8654
N = 30	0.9190	0.9118	0.9186	0.9170
N = 100	0.9376	0.9356	0.9402	0.9390



B PROP	L = 0.01	L = 0.1	L = 1	L = 10
N = 5	0.8890	0.9030	0.8926	0.8918
N = 10	0.9148	0.9272	0.9190	0.9278
N = 30	0.9396	0.9354	0.9424	0.9402
N = 100	0.9464	0.9498	0.9418	0.9460



2c)

For Graph 1 (Fixed Lambda and varying 'n')

- Graphs don't change much when λ is changed
- So the coverage probabilities don't depend on λ
- The coverage probabilities we get via bootstrap are higher than those of zinterval method.

For Graph 2 (Fixed 'n' and varying lambda)

- Graphs change when n is changed,
- The coverage probabilities depend on n.
- As the 'n' value increases (large samples), coverage probabilities of z-interval and bootstrap method are getting closer to each other

General:

- When n is large (n=100) for the bootstrap method coverage probabilities are on the higher side
- In case of bootstrap method, from n=30 itself we could observe larger coverage probabilities
- For every combination of (n, λ), coverage probability computed via bootstrap method is higher than z-interval method computed ones
- Hence, bootstrap method is accurate even for lower values of 'n'.
- Therefore, bootstrap method is more accurate and this is the recommended approach as well

2d)

The conclusions do not depend on lambda values. This is inferred from coverage probabilities values generated using both bootstrap method and z-interval method, we can see that irrespective of what the lambda values are, conclusions obtained in 2) c) are held.

SECTION - 2

QUESTION – 1

```
# reading data from csv file
bodytemp_heartrate = read.csv('C:/Aneena/bodytemp-heartrate.csv')
# creating subsets for men and women data
men = subset(bodytemp_heartrate, bodytemp_heartrate$gender == 1)
women = subset(bodytemp_heartrate, bodytemp_heartrate$gender == 2)
# Summary Statistics for body temperature
summary(men$body_temperature)
summary(women$body_temperature)
# Drawing boxplots for body temperature values
boxplot(men$body_temperature, women$body_temperature, main = "Boxplots of
Body Temperatures", names = c('MEN', 'WOMEN'), ylab = "Temperatures")
# Drawing Q-Q plots for the body temperature values
par(mfrow=c(1,2))
qqnorm(men$body_temperature, main = 'Q-Q Plot for Men')
qqline(men$body_temperature)
qqnorm(women$body_temperature, main = 'Q-Q Plot for Women')
qqline(women$body_temperature)
# confidence interval using t,test function for the body temperature values
t.test(men$body_temperature, women$body_temperature, alternative =
        'two.sided', var.equal = F)

# Summary Statistics for heart rate values
summary(men$heart_rate)
summary(women$heart_rate)
# Drawing boxplot for the heart rate values
boxplot(men$heart_rate, women$heart_rate, main = "Boxplots of Heart Rates",
       names = c('MEN', 'WOMEN'), ylab = "Heart Rates")
#drawing Q-Q plot for the heart rate values
par(mfrow=c(1,2))
qqnorm(men$heart_rate, main = 'Q-Q Plot for Men')
qqline(men$heart_rate)
qqnorm(women$heart_rate, main = 'Q-Q Plot for Women')
qqline(women$heart_rate)
# Confidence interval using the t.test function for the heart rate values
t.test(men$heart_rate, women$heart_rate, alternative = 'two.sided', var.equal = F)
# Finding the correlation values between body temperatures and heart rates
cor(men$body_temperature, men$heart_rate)
cor(women$body_temperature, women$heart_rate)
# Drawing the scatter plots for the body temperature & heart rate values for males & females
par(mfrow=c(1,2))
```

```
plot(men$heart_rate, men$body_temperature, pch=1, main='Scatter Plot for
Men')
abline(lm(men$body_temperature~men$heart_rate))
plot(women$heart_rate, women$body_temperature, pch=1, main='Scatter Plot for
Women')
abline(lm(women$body_temperature~women$heart_rate))
```

QUESTION – 2

```
# Creating check_z_ci function - checks whether true mean exists within confidence interval
# Simulating a sample and constructing an interval
check_z_ci <- function(n, lambda) {

  U <- rexp(n,lambda)
  lbound <- mean(U) - qnorm(0.975) * sd(U) / sqrt(n)
  ubound <- mean(U) + qnorm(0.975) * sd(U) / sqrt(n)
  tm = 1/lambda
  if(ubound>tm & lbound<tm) {
    return (1)
  }
  else {
    return (0)
  }
}

# Creating function zproportion - compute coverage probabilities using checkztmci
zproportion <- function(n, lambda) {
  values <- replicate(5000, check_z_ci(n, lambda))
  ones <- values[which (values == 1)]
  return (length(ones)/5000)
}

zproportion(5,0.01)

# Creating function mean_find - Returns mean of samples from distribution
mean_find <- function(n,lambda) {
  u.star <- rexp(n, lambda)
  return (mean(u.star))
}

# Creating check_b_ci function - checks whether true mean exists within confidence interval
# calls the mean_find func() 1000 times and forms the confidence interval

check_b_ci <- function(n, lambda) {
```

```
  U <- rexp(n,lambda)
  tm <- 1/lambda
  lambda1 = 1/mean(U)
  V <- replicate(1000, mean_find(n,lambda1))
  bound <- sort(V)[c(25, 975)]
  if(bound[2]>tm & bound[1]<tm) {
    return (1)
  }
  else {
    return (0)
  }
}

# Creating function bproportion - constructs a parametric initial bootstrap sample
# calls checkbtmci 5000 times and calculates coverage probabilities

bproportion <- function(n, lambda) {
  values <- replicate(5000, check_b_ci(n, lambda))
  ones <- values[which (values == 1)]
  return (length(ones)/5000)
}

bproportion(5,0.01)

# Generating the proportion values for bootstrap and z-interval
zcimatrix <- matrix(c(zproportion(5,0.01), zproportion(10,0.01),
                      zproportion(30,0.01), zproportion(100,0.01), zproportion(5,0.1), zproportion(10,0.1),
                      zproportion(30,0.1), zproportion(100,0.1), zproportion(5,1), zproportion(10,1),
                      zproportion(30,1), zproportion(100,1), zproportion(5,10), zproportion(10,10),
                      zproportion(30,10), zproportion(100,10)), nrow = 4, ncol = 4)
zcimatrix

bcimatrix <- matrix(c(bproportion(5,0.01), bproportion(10,0.01),
                      bproportion(30,0.01), bproportion(100,0.01), bproportion(5,0.1), bproportion(10,0.1),
                      bproportion(30,0.1), bproportion(100,0.1), bproportion(5,1), bproportion(10,1),
                      bproportion(30,1), bproportion(100,1), bproportion(5,10), bproportion(10,10),
                      bproportion(30,10), bproportion(100,10)), nrow = 4, ncol = 4)

bcimatrix

# Drawing line graphs for all these values
par(mfrow=c(2,2))

# Fixed lambda and varying 'n'
```

```
plot(c(5,10,30,100), zcimatrix[,1], main = "L = 0.01", xlab = 'n', ylab =
      'Proportions', col = 'red', type = 'b', xlim = c(1,100), ylim = c(0.8,1))
lines(c(5,10,30,100), bcimatrix[,1], col = 'blue', type = 'b')
plot(c(5,10,30,100), zcimatrix[,2], main = "L = 0.1", xlab = 'n', ylab = 'Proportions',
    col = 'red', type = 'b', xlim = c(1,100), ylim = c(0.8,1))
lines(c(5,10,30,100), bcimatrix[,2], col = 'blue', type = 'b')
plot(c(5,10,30,100), zcimatrix[,3], main = "L = 1", xlab = 'n', ylab = 'Proportions',
    col = 'red', type = 'b', xlim = c(1,100), ylim = c(0.8,1))
lines(c(5,10,30,100), bcimatrix[,3], col = 'blue', type = 'b')
plot(c(5,10,30,100), zcimatrix[,4], main = "L = 10", xlab = 'n', ylab = 'Proportions',
    col = 'red', type = 'b', xlim = c(1,100), ylim = c(0.8,1))
lines(c(5,10,30,100), bcimatrix[,4], col = 'blue', type = 'b')

# Fixed 'n' and varying lambda

plot(c(0.01,0.1,1,10), zcimatrix[1,], main = "N = 5", xlab = 'Lambda', ylab =
      'Proportions', col = 'red', type = 'b', xlim = c(0.01,10), ylim = c(0,1))
lines(c(0.01,0.1,1,10), bcimatrix[1,], col = 'blue', type = 'b')
plot(c(0.01,0.1,1,10), zcimatrix[2,], main = "N = 10", xlab = 'Lambda', ylab =
      'Proportions', col = 'red', type = 'b', xlim = c(0.01,10), ylim = c(0,1))
lines(c(0.01,0.1,1,10), bcimatrix[2,], col = 'blue', type = 'b')
plot(c(0.01,0.1,1,10), zcimatrix[3,], main = "N = 30", xlab = 'Lambda', ylab =
      'Proportions', col = 'red', type = 'b', xlim = c(0.01,10), ylim = c(0,1))
lines(c(0.01,0.1,1,10), bcimatrix[3,], col = 'blue', type = 'b')
plot(c(0.01,0.1,1,10), zcimatrix[4,], main = "N = 100", xlab = 'Lambda', ylab =
      'Proportions', col = 'red', type = 'b', xlim = c(0.01,10), ylim = c(0,1))
lines(c(0.01,0.1,1,10), bcimatrix[4,], col = 'blue', type = 'b')
```