Homework 2 – Due February 17

Name: Navya Mittal netID: navya_0215 Collaborated with:No one

Your homework **must be submitted in Word or PDF format, created by calling "Knit Word" or "Knit PDF" from RStudio on your R Markdown document. Submission in other formats may receive a grade of 0**. Your responses must be supported by both textual explanations and the code you generate to produce your result. Note that all R code used to produce your results must be shown in your knitted file. You can collaborate with your classmates, but you must identify their names above, and you must submit your own homework as a knitted file.

Question 1

'airquality' is a dataset recording daily air quality measurements in New York. Write a R program to

(a) show the first 5 rows of this data

```
data(airquality)
head(airquality)
```

```
##   Ozone Solar.R Wind Temp Month Day
## 1    41     190  7.4   67     5   1
## 2    36     118  8.0   72     5   2
## 3    12     149 12.6   74     5   3
## 4    18     313 11.5   62     5   4
## 5    NA      NA 14.3   56     5   5
## 6    28      NA 14.9   66     5   6
```

(b) count the number of missing values in each column

```
sum(is.na(airquality$Ozone))
```

```
## [1] 37
```

```
sum(is.na(airquality$Solar.R))
```

```
## [1] 7
```

```
sum(is.na(airquality$Wind))
```

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

```
sum(is.na(airquality$Temp))
```

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

```
sum(is.na(airquality$Month))
```

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

```
sum(is.na(airquality$Day))
```

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

(c) use apply function to calculate the maximum for each column (excluding missing values).

```
# ?apply
# dim(airquality)
apply(airquality, 2, max, na.rm = TRUE)
```

```
##   Ozone Solar.R    Wind    Temp   Month     Day
##   168.0   334.0    20.7    97.0     9.0    31.0
```

(d) extract all observations (a subset of dataframe) for the month of May.

```
may <- subset(airquality, Month == "5")
may
```

```
##    Ozone Solar.R Wind Temp Month Day
## 1     41     190  7.4   67     5   1
## 2     36     118  8.0   72     5   2
## 3     12     149 12.6   74     5   3
## 4     18     313 11.5   62     5   4
## 5     NA      NA 14.3   56     5   5
## 6     28      NA 14.9   66     5   6
## 7     23     299  8.6   65     5   7
## 8     19      99 13.8   59     5   8
## 9      8      19 20.1   61     5   9
## 10    NA     194  8.6   69     5  10
## 11     7      NA  6.9   74     5  11
## 12    16     256  9.7   69     5  12
## 13    11     290  9.2   66     5  13
## 14    14     274 10.9   68     5  14
## 15    18      65 13.2   58     5  15
## 16    14     334 11.5   64     5  16
## 17    34     307 12.0   66     5  17
## 18     6      78 18.4   57     5  18
## 19    30     322 11.5   68     5  19
## 20    11      44  9.7   62     5  20
## 21     1       8  9.7   59     5  21
## 22    11     320 16.6   73     5  22
## 23     4      25  9.7   61     5  23
## 24    32      92 12.0   61     5  24
## 25    NA      66 16.6   57     5  25
## 26    NA     266 14.9   58     5  26
## 27    NA      NA  8.0   57     5  27
## 28    23      13 12.0   67     5  28
## 29    45     252 14.9   81     5  29
## 30   115     223  5.7   79     5  30
## 31    37     279  7.4   76     5  31
```

(e) extract all rows (a subset of dataframe) where 'Ozone' is less than its 1st quartile and 'Month' is not May.

```
ozonenotmay <- subset(airquality, Month!= "5" & Ozone < 25)
ozonenotmay
```

```
##     Ozone Solar.R Wind Temp Month Day
## 44     23     148  8.0   82     6  13
## 47     21     191 14.9   77     6  16
## 49     20      37  9.2   65     6  18
## 50     12     120 11.5   73     6  19
```

```
## 51     13     137 10.3   76     6  20
## 73     10     264 14.3   73     7  12
## 76      7      48 14.3   80     7  15
## 82     16       7  6.9   74     7  21
## 87     20      81  8.6   82     7  26
## 94      9      24 13.8   81     8   2
## 95     16      77  7.4   82     8   3
## 108    22      71 10.3   77     8  16
## 110    23     115  7.4   76     8  18
## 113    21     259 15.5   77     8  21
## 114     9      36 14.3   72     8  22
## 130    20     252 10.9   80     9   7
## 131    23     220 10.3   78     9   8
## 132    21     230 10.9   75     9   9
## 133    24     259  9.7   73     9  10
## 135    21     259 15.5   76     9  12
## 137     9      24 10.9   71     9  14
## 138    13     112 11.5   71     9  15
## 140    18     224 13.8   67     9  17
## 141    13      27 10.3   76     9  18
## 142    24     238 10.3   68     9  19
## 143    16     201  8.0   82     9  20
## 144    13     238 12.6   64     9  21
## 145    23      14  9.2   71     9  22
## 147     7      49 10.3   69     9  24
## 148    14      20 16.6   63     9  25
## 151    14     191 14.3   75     9  28
## 152    18     131  8.0   76     9  29
## 153    20     223 11.5   68     9  30
```

(f) 'Temp' is in degrees Fahrenheit. Add a new column called 'Celsius' by transforming 'Temp' to be in degrees Celsius. Note: Celsius=(Fahrenheit-32)/1.8.

```
airquality["Celsius"] = (airquality$Temp -32)/1.8
airquality
```

```
##     Ozone Solar.R Wind Temp Month Day  Celsius
## 1      41     190  7.4   67     5   1 19.44444
## 2      36     118  8.0   72     5   2 22.22222
## 3      12     149 12.6   74     5   3 23.33333
## 4      18     313 11.5   62     5   4 16.66667
## 5      NA      NA 14.3   56     5   5 13.33333
## 6      28      NA 14.9   66     5   6 18.88889
## 7      23     299  8.6   65     5   7 18.33333
## 8      19      99 13.8   59     5   8 15.00000
## 9       8      19 20.1   61     5   9 16.11111
## 10     NA     194  8.6   69     5  10 20.55556
## 11      7      NA  6.9   74     5  11 23.33333
## 12     16     256  9.7   69     5  12 20.55556
## 13     11     290  9.2   66     5  13 18.88889
## 14     14     274 10.9   68     5  14 20.00000
## 15     18      65 13.2   58     5  15 14.44444
## 16     14     334 11.5   64     5  16 17.77778
## 17     34     307 12.0   66     5  17 18.88889
## 18      6      78 18.4   57     5  18 13.88889
```

```
## 19     30     322 11.5   68     5  19 20.00000
## 20     11      44  9.7   62     5  20 16.66667
## 21      1       8  9.7   59     5  21 15.00000
## 22     11     320 16.6   73     5  22 22.77778
## 23      4      25  9.7   61     5  23 16.11111
## 24     32      92 12.0   61     5  24 16.11111
## 25     NA      66 16.6   57     5  25 13.88889
## 26     NA     266 14.9   58     5  26 14.44444
## 27     NA      NA  8.0   57     5  27 13.88889
## 28     23      13 12.0   67     5  28 19.44444
## 29     45     252 14.9   81     5  29 27.22222
## 30    115     223  5.7   79     5  30 26.11111
## 31     37     279  7.4   76     5  31 24.44444
## 32     NA     286  8.6   78     6   1 25.55556
## 33     NA     287  9.7   74     6   2 23.33333
## 34     NA     242 16.1   67     6   3 19.44444
## 35     NA     186  9.2   84     6   4 28.88889
## 36     NA     220  8.6   85     6   5 29.44444
## 37     NA     264 14.3   79     6   6 26.11111
## 38     29     127  9.7   82     6   7 27.77778
## 39     NA     273  6.9   87     6   8 30.55556
## 40     71     291 13.8   90     6   9 32.22222
## 41     39     323 11.5   87     6  10 30.55556
## 42     NA     259 10.9   93     6  11 33.88889
## 43     NA     250  9.2   92     6  12 33.33333
## 44     23     148  8.0   82     6  13 27.77778
## 45     NA     332 13.8   80     6  14 26.66667
## 46     NA     322 11.5   79     6  15 26.11111
## 47     21     191 14.9   77     6  16 25.00000
## 48     37     284 20.7   72     6  17 22.22222
## 49     20      37  9.2   65     6  18 18.33333
## 50     12     120 11.5   73     6  19 22.77778
## 51     13     137 10.3   76     6  20 24.44444
## 52     NA     150  6.3   77     6  21 25.00000
## 53     NA      59  1.7   76     6  22 24.44444
## 54     NA      91  4.6   76     6  23 24.44444
## 55     NA     250  6.3   76     6  24 24.44444
## 56     NA     135  8.0   75     6  25 23.88889
## 57     NA     127  8.0   78     6  26 25.55556
## 58     NA      47 10.3   73     6  27 22.77778
## 59     NA      98 11.5   80     6  28 26.66667
## 60     NA      31 14.9   77     6  29 25.00000
## 61     NA     138  8.0   83     6  30 28.33333
## 62    135     269  4.1   84     7   1 28.88889
## 63     49     248  9.2   85     7   2 29.44444
## 64     32     236  9.2   81     7   3 27.22222
## 65     NA     101 10.9   84     7   4 28.88889
## 66     64     175  4.6   83     7   5 28.33333
## 67     40     314 10.9   83     7   6 28.33333
## 68     77     276  5.1   88     7   7 31.11111
## 69     97     267  6.3   92     7   8 33.33333
## 70     97     272  5.7   92     7   9 33.33333
## 71     85     175  7.4   89     7  10 31.66667
## 72     NA     139  8.6   82     7  11 27.77778
```

```
## 73     10     264 14.3   73     7  12 22.77778
## 74     27     175 14.9   81     7  13 27.22222
## 75     NA     291 14.9   91     7  14 32.77778
## 76      7      48 14.3   80     7  15 26.66667
## 77     48     260  6.9   81     7  16 27.22222
## 78     35     274 10.3   82     7  17 27.77778
## 79     61     285  6.3   84     7  18 28.88889
## 80     79     187  5.1   87     7  19 30.55556
## 81     63     220 11.5   85     7  20 29.44444
## 82     16       7  6.9   74     7  21 23.33333
## 83     NA     258  9.7   81     7  22 27.22222
## 84     NA     295 11.5   82     7  23 27.77778
## 85     80     294  8.6   86     7  24 30.00000
## 86    108     223  8.0   85     7  25 29.44444
## 87     20      81  8.6   82     7  26 27.77778
## 88     52      82 12.0   86     7  27 30.00000
## 89     82     213  7.4   88     7  28 31.11111
## 90     50     275  7.4   86     7  29 30.00000
## 91     64     253  7.4   83     7  30 28.33333
## 92     59     254  9.2   81     7  31 27.22222
## 93     39      83  6.9   81     8   1 27.22222
## 94      9      24 13.8   81     8   2 27.22222
## 95     16      77  7.4   82     8   3 27.77778
## 96     78      NA  6.9   86     8   4 30.00000
## 97     35      NA  7.4   85     8   5 29.44444
## 98     66      NA  4.6   87     8   6 30.55556
## 99    122     255  4.0   89     8   7 31.66667
## 100    89     229 10.3   90     8   8 32.22222
## 101   110     207  8.0   90     8   9 32.22222
## 102    NA     222  8.6   92     8  10 33.33333
## 103    NA     137 11.5   86     8  11 30.00000
## 104    44     192 11.5   86     8  12 30.00000
## 105    28     273 11.5   82     8  13 27.77778
## 106    65     157  9.7   80     8  14 26.66667
## 107    NA      64 11.5   79     8  15 26.11111
## 108    22      71 10.3   77     8  16 25.00000
## 109    59      51  6.3   79     8  17 26.11111
## 110    23     115  7.4   76     8  18 24.44444
## 111    31     244 10.9   78     8  19 25.55556
## 112    44     190 10.3   78     8  20 25.55556
## 113    21     259 15.5   77     8  21 25.00000
## 114     9      36 14.3   72     8  22 22.22222
## 115    NA     255 12.6   75     8  23 23.88889
## 116    45     212  9.7   79     8  24 26.11111
## 117   168     238  3.4   81     8  25 27.22222
## 118    73     215  8.0   86     8  26 30.00000
## 119    NA     153  5.7   88     8  27 31.11111
## 120    76     203  9.7   97     8  28 36.11111
## 121   118     225  2.3   94     8  29 34.44444
## 122    84     237  6.3   96     8  30 35.55556
## 123    85     188  6.3   94     8  31 34.44444
## 124    96     167  6.9   91     9   1 32.77778
## 125    78     197  5.1   92     9   2 33.33333
## 126    73     183  2.8   93     9   3 33.88889
```

```
## 127    91     189  4.6   93     9   4 33.88889
## 128    47      95  7.4   87     9   5 30.55556
## 129    32      92 15.5   84     9   6 28.88889
## 130    20     252 10.9   80     9   7 26.66667
## 131    23     220 10.3   78     9   8 25.55556
## 132    21     230 10.9   75     9   9 23.88889
## 133    24     259  9.7   73     9  10 22.77778
## 134    44     236 14.9   81     9  11 27.22222
## 135    21     259 15.5   76     9  12 24.44444
## 136    28     238  6.3   77     9  13 25.00000
## 137     9      24 10.9   71     9  14 21.66667
## 138    13     112 11.5   71     9  15 21.66667
## 139    46     237  6.9   78     9  16 25.55556
## 140    18     224 13.8   67     9  17 19.44444
## 141    13      27 10.3   76     9  18 24.44444
## 142    24     238 10.3   68     9  19 20.00000
## 143    16     201  8.0   82     9  20 27.77778
## 144    13     238 12.6   64     9  21 17.77778
## 145    23      14  9.2   71     9  22 21.66667
## 146    36     139 10.3   81     9  23 27.22222
## 147     7      49 10.3   69     9  24 20.55556
## 148    14      20 16.6   63     9  25 17.22222
## 149    30     193  6.9   70     9  26 21.11111
## 150    NA     145 13.2   77     9  27 25.00000
## 151    14     191 14.3   75     9  28 23.88889
## 152    18     131  8.0   76     9  29 24.44444
## 153    20     223 11.5   68     9  30 20.00000
```

(g) Find the mean temp for each month.

```
for (i in 5:9){
  print(mean(airquality[(airquality$Month== i), "Temp"]))
}
```

```
## [1] 65.54839
## [1] 79.1
## [1] 83.90323
## [1] 83.96774
## [1] 76.9
```

The first value is the mean temperature for May, second value is the mean temperature for June and so on. I used the range 5:9 because that's the range of months included in the database. ## Question 2

The Fibonacci numbers are the sequence of numbers defined by the linear recurrence equation F(n) = F(n-1) + F(n-2) , where F(1) = F(2) = 1 and by convention F(0) = 0. For example, the first 8 Fibonacci numbers are 1, 1, 2, 3, 5, 8, 13, 21.

(a) write your own R function to compute the n-th Fibonnaci number. Use this function to calculate F(n=15);

```
fib <- function(n) {
  if (n <= 0) {
    return(0)
  }
  if (n == 1) {
    return(1)
  }
```

```
  return (fib(n - 1) + fib(n - 2))
}

fib(15)
```

```
## [1] 610
```

(b) Write you own R function to compute a sequence of Fibonacci numbers such that the last element in the sequence is less than K. Use this function and a `while()/break` combination to find the sequence F(0),F(1),...,F(n) such that F(n)<500;

```
fib_K <- function(K) {
  fib_seq <- numeric(2)
  fib_seq[1] <- 0
  fib_seq[2] <- 1
  n <- 2
  while (TRUE) {
    next_fib <- fib_seq[n - 1] + fib_seq[n]
    if (next_fib >= K) {
      break
    }
    n <- n + 1
    fib_seq[n] <- next_fib
  }
  return(fib_seq[1:n])
}

fib_500 <- fib_K(500)
fib_500
```

```
##  [1]   0   1   1   2   3   5   8  13  21  34  55  89 144 233 377
```

Question 3

In STAT211, you have learned that given a sample of size n from a normal distribution, the CL=95% confidence interval for the mean can be calculated by

$$\bar{x} \pm z_{(1-CL)/2} * s/\sqrt{n}.$$

Where $z_{(1-CL)/2)} = z(.025)$ is the z multiplier.

(a) help(qnorm) function. Use qnorm(1-.025) to find z(.025).

```
?qnorm
qnorm(1-.025)
```

```
## [1] 1.959964
```

That is the value of z(0.25).

(b) Create a vector x by generating $n = 50$ numbers from N(mean=30,sd=2) distribution. Calculate the confidence interval from this data using the CI formula. Check whether the interval covers the true mean=30 or not.

```
x <- rnorm(50, mean = 30, sd = 2)
xmean <- mean(x)
stdev <- sd(x)
ci <- 0.95
```

```r
alpha <- 1 - ci
df <- length(x) - 1
t <- qt(1 - alpha/2, df)
lower <- xmean - t * stdev / sqrt(length(x))
upper <- xmean + t * stdev / sqrt(length(x))
lower
```

```
## [1] 29.26596
```

```r
upper
```

```
## [1] 30.4493
```

```r
if (30 >= lower && 30 <= upper) {
  print("The true mean of 30 is covered by CI.")
} else {
  print("The true mean of 30 is not covered by CI.")
}
```

```
## [1] "The true mean of 30 is covered by CI."
```

(c) Repeat the above experiments for 200 times to obtain 200 such intervals using a `for()` loop. Calculate the percentage of intervals that cover the true mean=30. This is the empirical coverage probability. In theory, it should be very close to your CL.

```r
count = 0
for(i in 1:200){
  x <- rnorm(50, mean = 30, sd = 2)
  xmean <- mean(x)
  stdev <- sd(x)
  ci <- 0.95
  alpha <- 1 - ci
  df <- length(x) - 1
  t <- qt(1 - alpha/2, df)
  lower <- xmean - t * stdev / sqrt(length(x))
  upper <- xmean + t * stdev / sqrt(length(x))

  if (30 >= lower && 30 <= upper) {
    count <- count + 1
  }
}

emp <- count / 200
emp
```

```
## [1] 0.93
```

(d) Write a function using CL as an input argument, and the percentage calculated from question c as an output. Use this function to create a 5 by 2 matrix with one column showing the theoretical CL and the other showing the empirical coverage probability, for CL=.8, .85, .9, .95,.99.

```r
ci_coverage <- function(ci) {
  count = 0
  for (i in 1:200) {
    x <- rnorm(50, mean = 30, sd = 2)
    xmean <- mean(x)
    stdev <- sd(x)
    alpha <- 1 - ci
```

```
    df <- length(x) - 1
    t <- qt(1 - alpha/2, df)
    lower <- xmean - t * stdev / sqrt(length(x))
    upper <- xmean + t * stdev / sqrt(length(x))

    if (30 >= lower && 30 <= upper) {
      count = count + 1
    }
  }
  emp <- count / 200
  return(emp)
}

cl_levels <- c(0.8, 0.85, 0.9, 0.95, 0.99)
results <- matrix(0, nrow = length(cl_levels), ncol = 2)
colnames(results) <- c("Theoretical CL", "Empirical Coverage Probability")
for (i in 1:length(cl_levels)) {
  results[i, 1] <- cl_levels[i]
  results[i, 2] <- ci_coverage(cl_levels[i])
}
results
```

```
##      Theoretical CL Empirical Coverage Probability
## [1,]           0.80                           0.810
## [2,]           0.85                           0.865
## [3,]           0.90                           0.905
## [4,]           0.95                           0.965
## [5,]           0.99                           0.995
```