Section 1: R Programming Questions: ######################################

Q.1.:->

We have been provided with following information in the question: A clinic has three doctors. Patients come into the clinic at random, starting at 9 a.m. The time after opening at which the first patient appears follows an exponential distribution with expectation of 10 minutes and then, after each patient arrives, the waiting time until the next patient is independently exponentially distributed, also with expectation of 10 minutes. When a patient arrives, he or she waits until a doctor is available. The amount of time spent by each doctor with each patient is a random variable, uniformly distributed between 5 and 20 minutes. The office stops admitting new patients at 4 p.m. and closes when the last patient is through with the doctor.

Q.1.(a)

In this part of question, we need to simulate the above process once.

```
#setting a random seed value of 243 to get same/ reproducible results every
time we run this code
set.seed(243)

# expected arrival waiting time = 10 mins
# and arrivals are following an independent exponential distribution
#therefore, (1/lambda) = 10,
#therefore, lambda = 0.1

#therefore, we can estimate the arrival times of patient during these total
of 420 mins (between 9 a.m. and 4 p.m.)
#for this, we will use a built-in function REXP in R for random exponential
numbers generation with lambda = 0.1

AT <- rexp(100, rate = 0.1)

#calculating cumulative arrival times
CAT <- cumsum(AT)

#from this cumulative sum vector "CAT", we will find number of patients who
arrived before 420 minutes or 4 p.m.
A <- sum(CAT<420)
A

## [1] 32

#so , from the result of vector "A", we can observe that in this case, with
set.seed(243), 32 number of patients arrived before 420 minutes or 4 p.m.
# This gives us answer to part (i) of our question
```

```r
########################################################################

#now, we will generate data regarding amount of time spent by each patient during consultation with the doctor
DT <- runif(A, min = 5, max = 20)

#creating a matrix to determine how each patient will be assigned with a doctor and how much will be the wait time

#first creating a matrix of NA values
W_MAT <- matrix(NA, nrow = A, ncol = 10)

#assigning names to all columns of the matrix
colnames(W_MAT) <- c("Arr Time", "Dr6/7/8", "Starts", "Dur", "Ends", "Dr6", "Dr7", "Dr8", "W", "WT")

#note that we have designated 3 doctors of the clinic with Dr6, Dr7 and Dr8 names respectively as they are placed in the 6th, 7th and 8th column of the matrix

#we will assign doctor Dr6/7/8 to patient such that:
#(a) if all doctors are busy, he need to wait for a doctor only until any one doctor becomes available
#(b) if multiple doctors are available for a patient, he will go to the doctor who was available from the earliest time frame

#for the first patient (i.e. first row), assigning values for all respective columns 1 to 10 named as "Arr Time", "Dr6/7/8", "Starts", "Dur", "Ends", "Dr6", "Dr7", "Dr8", "W", "WT"
W_MAT[1,1] <- CAT[1]
W_MAT[1,2] <- 6
W_MAT[1,3] <- W_MAT[1,1]
W_MAT[1,4] <- DT[1]
W_MAT[1,5] <- sum(W_MAT[1,3],W_MAT[1,4])
W_MAT[1,6] <- W_MAT[1,5]
W_MAT[1,7] <- 0
W_MAT[1,8] <- 0
W_MAT[1,9] <- 0
W_MAT[1,10] <- 0

#now assigning other patient data in similar way with above stated concepts
for(i in 2:A){

  #using cumulative arrival times of patients
  W_MAT[i,1] <- CAT[i]

  #we assign doctor Dr6/7/8 to patient such that:
  #(a) if all doctors are busy, he need to wait for a doctor only until any one doctor becomes available
```

```
  #(b) if multiple doctors are available for a patient, he will go to the 
doctor who was available from the earliest time frame
  W_MAT[i,2] <- sum(5,which.min(c(W_MAT[i-1,6],W_MAT[i-1,7],W_MAT[i-1,8])))

  if(W_MAT[i,1] < W_MAT[i-1,W_MAT[i,2]]){
    W_MAT[i,3] <- W_MAT[i-1,W_MAT[i,2]]
  }else{
    W_MAT[i,3] <- W_MAT[i,1]
  }

  #using time spent by patients with doctor
  W_MAT[i,4] <- DT[i]

  #exit time of that patient from clinic
  W_MAT[i,5] <- sum(W_MAT[i,3],W_MAT[i,4])

  #updating Dr6 occupancy timestamp if this patient is in consultation with 
doctor
  if(W_MAT[i,2] == 6){
    W_MAT[i,6] <- W_MAT[i,5]
  }else{
    W_MAT[i,6] <- W_MAT[i-1,6]
  }

  #updating Dr7 occupancy timestamp if this patient is in consultation with 
doctor
  if(W_MAT[i,2] == 7){
    W_MAT[i,7] <- W_MAT[i,5]
  }else{
    W_MAT[i,7] <- W_MAT[i-1,7]
  }

  #updating Dr8 occupancy timestamp if this patient is in consultation with 
doctor
  if(W_MAT[i,2] == 8){
    W_MAT[i,8] <- W_MAT[i,5]
  }else{
    W_MAT[i,8] <- W_MAT[i-1,8]
  }

  #determining if this patient had to wait or not
  W_MAT[i,9] <- (W_MAT[i,"Arr Time"] < W_MAT[i,"Starts"]) * 1

  #determining wait time if this patient had to wait
  W_MAT[i,10] <- (W_MAT[i,"Starts"]-W_MAT[i,"Arr Time"]) * W_MAT[i,9]

}
```

```
#so now we can see the matrix with all updated data in it
W_MAT

##         Arr Time Dr6/7/8    Starts       Dur      Ends       Dr6       Dr7
##  [1,]  11.73359       6  11.73359  8.075296  19.80888  19.80888   0.00000
##  [2,]  20.39953       7  20.39953 15.286732  35.68626  19.80888  35.68626
##  [3,]  25.66015       8  25.66015  7.571070  33.23122  19.80888  35.68626
##  [4,]  31.29808       6  31.29808 15.449553  46.74764  46.74764  35.68626
##  [5,]  53.49361       8  53.49361 11.638936  65.13255  46.74764  35.68626
##  [6,]  67.16264       7  67.16264 14.496379  81.65902  46.74764  81.65902
##  [7,]  79.27071       6  79.27071 19.121291  98.39200  98.39200  81.65902
##  [8,]  84.98348       8  84.98348 12.319853  97.30333  98.39200  81.65902
##  [9,]  85.01312       7  85.01312 16.079053 101.09218  98.39200 101.09218
## [10,] 114.27862       8 114.27862  7.585585 121.86421  98.39200 101.09218
## [11,] 127.24158       6 127.24158  7.807285 135.04886 135.04886 101.09218
## [12,] 140.48698       7 140.48698 18.269386 158.75637 135.04886 158.75637
## [13,] 146.48946       8 146.48946 15.179458 161.66892 135.04886 158.75637
## [14,] 149.70361       6 149.70361 10.993399 160.69701 160.69701 158.75637
## [15,] 152.13597       7 158.75637 11.419941 170.17631 160.69701 170.17631
## [16,] 162.24503       6 162.24503 17.920499 180.16553 180.16553 170.17631
## [17,] 170.03512       8 170.03512 13.314208 183.34933 180.16553 170.17631
## [18,] 170.52518       7 170.52518  8.979134 179.50431 180.16553 179.50431
## [19,] 176.99011       7 179.50431 15.048659 194.55297 180.16553 194.55297
## [20,] 216.54503       6 216.54503 14.694457 231.23949 231.23949 194.55297
## [21,] 224.75679       8 224.75679 19.472750 244.22954 231.23949 194.55297
## [22,] 254.48005       7 254.48005 13.555716 268.03576 231.23949 268.03576
## [23,] 260.07441       6 260.07441 11.168827 271.24324 271.24324 268.03576
## [24,] 281.44525       8 281.44525 14.408563 295.85382 271.24324 268.03576
## [25,] 330.46214       7 330.46214 13.323931 343.78607 271.24324 343.78607
## [26,] 354.79602       6 354.79602  8.192454 362.98847 362.98847 343.78607
## [27,] 365.85396       8 365.85396  9.006508 374.86047 362.98847 343.78607
## [28,] 374.73370       7 374.73370 18.247882 392.98158 362.98847 392.98158
## [29,] 384.27567       6 384.27567  8.863893 393.13956 393.13956 392.98158
## [30,] 398.26934       8 398.26934 13.312021 411.58136 393.13956 392.98158
## [31,] 402.35521       7 402.35521 13.906389 416.26160 393.13956 416.26160
## [32,] 417.56405       6 417.56405 14.929635 432.49369 432.49369 416.26160
##             Dr8 W       WT
##  [1,]   0.00000 0 0.000000
##  [2,]   0.00000 0 0.000000
##  [3,]  33.23122 0 0.000000
##  [4,]  33.23122 0 0.000000
##  [5,]  65.13255 0 0.000000
##  [6,]  65.13255 0 0.000000
##  [7,]  65.13255 0 0.000000
##  [8,]  97.30333 0 0.000000
##  [9,]  97.30333 0 0.000000
## [10,] 121.86421 0 0.000000
## [11,] 121.86421 0 0.000000
## [12,] 121.86421 0 0.000000
## [13,] 161.66892 0 0.000000
```

```
## [14,] 161.66892 0 0.000000
## [15,] 161.66892 1 6.620393
## [16,] 161.66892 0 0.000000
## [17,] 183.34933 0 0.000000
## [18,] 183.34933 0 0.000000
## [19,] 183.34933 1 2.514206
## [20,] 183.34933 0 0.000000
## [21,] 244.22954 0 0.000000
## [22,] 244.22954 0 0.000000
## [23,] 244.22954 0 0.000000
## [24,] 295.85382 0 0.000000
## [25,] 295.85382 0 0.000000
## [26,] 295.85382 0 0.000000
## [27,] 374.86047 0 0.000000
## [28,] 374.86047 0 0.000000
## [29,] 374.86047 0 0.000000
## [30,] 411.58136 0 0.000000
## [31,] 411.58136 0 0.000000
## [32,] 411.58136 0 0.000000

# now, using this "W_MAT" matrix results, we will find How many patients had
to wait for a doctor
#for this we will take a sum of values in "W" column of our matrix
A2 <- sum(W_MAT[,"W"])
A2

## [1] 2

#so , from the result of "A2", we can observe that in this case, with
set.seed(243), only 2 patients had to wait for a doctor
# This gives us answer to part (ii) of our question

######################################################################

# now, using this "W_MAT" matrix results, we will find what was patients
average wait time
#for this we will take average of values in "WT" column of our matrix
A3 <- sum(W_MAT[,"WT"])/ A
A3

## [1] 0.2854562

#so , from the result of "A3", we can observe that in this case, with
set.seed(243), patients average wait time was only 0.2854562 minutes
# This gives us answer to part (iii) of our question

######################################################################

# now, using this "W_MAT" matrix results, we will find when did the office
close or when the last patient is through with the doctor
```

```
#for this, we will use the maximum of doctors time-stamps values in the last row of the matrix
#because it is possible that patient admitted earlier than last patient may leave clinic last due to his higher consultation time with doctor, so it is better to find at what time all doctors complete their consultation and office can be closed
#also we will ensure that in cases where this value is less than 420 min, our answer should be 420 min and not anything less than that, because clinic can't be closed before 420 min or 4 p.m. (even though all existing patients completes their consultation with doctors before 4 p.m. and we are expecting no one to arrive before 4 pm as per out predicted arrival time)

A4 <- max(max(W_MAT[A,c("Dr6","Dr7","Dr8")]), 420)
A4

## [1] 432.4937

#this A4 gives us answer in minutes starting from 9 a.m.

#we can convert this A4 minutes to clock time in following way, while doing so we will be rounding of a fraction of minute to higher side by applying ceiling() function
A4TH <- cat(floor((A4/60)-3),":",ceiling(A4%%60),"p.m.")

## 4 : 13 p.m.

#so , from the result of "A4TH", we can observe that in this case, with set.seed(243), the office will be closed at 4:13 p.m.
# This gives us answer to part (iv) of our question

######################################################################
```

so, as derived and specified in the above code, considering set.seed(243) arbitrarily, we have got the following results for the questions asked in Q.1.(a):

(i) How many patients came to the office? Ans. 32

(ii) How many had to wait for a doctor? Ans. 2

(iii) What was their average wait? Ans. (0.2854562) minutes

(iv) When did the office close? Ans. 4:13 p.m.

Q.1.(b)

In this part of question, we need to simulate the above process 1000 times and estimate the median for each of the summaries in Q.1.(a)

```r
# creating empty vectors and a list which we will fill with values we receive in each simulation
A1 <- c()
A2 <- c()
A3 <- c()
A4 <- c()
All_W_MAT <- list()

#as asked in the question, simulating the process done in Q.1.(a) above, 1000 times
for(j in 1:1000){

  #using random seed value of 1 to 1000 so that our results stays reproducible for this experiment of 1000 simulations
  set.seed(j)

  ##########
  # NOTE: just repeating same code as used and explained above in Q.1.(a), so no need to explain all lines of code again
  ##########

  AT <- rexp(100, rate = 0.1)
  CAT <- cumsum(AT)
  A <- sum(CAT<420)

  A1[j] <- A
  DT <- runif(A, min = 5, max = 20)
  W_MAT <- matrix(NA, nrow = A, ncol = 10)
  colnames(W_MAT) <- c("Arr Time", "Dr6/7/8", "Starts", "Dur", "Ends", "Dr6", "Dr7", "Dr8", "W", "WT")

  W_MAT[1,1] <- CAT[1]
  W_MAT[1,2] <- 6
  W_MAT[1,3] <- W_MAT[1,1]
  W_MAT[1,4] <- DT[1]
  W_MAT[1,5] <- sum(W_MAT[1,3],W_MAT[1,4])
  W_MAT[1,6] <- W_MAT[1,5]
  W_MAT[1,7] <- 0
  W_MAT[1,8] <- 0
  W_MAT[1,9] <- 0
  W_MAT[1,10] <- 0

  ##########
  # NOTE: just repeating same code as used and explained above in Q.1.(a), so no need to explain all lines of code again
  ##########

  for(i in 2:A){
```

```
    W_MAT[i,1] <- CAT[i]
    W_MAT[i,2] <- sum(5,which.min(c(W_MAT[i-1,6],W_MAT[i-1,7],W_MAT[i-1,8])))

    if(W_MAT[i,1] < W_MAT[i-1,W_MAT[i,2]]){
      W_MAT[i,3] <- W_MAT[i-1,W_MAT[i,2]]
    }else{
      W_MAT[i,3] <- W_MAT[i,1]
    }

    #using time spent by patients with doctor
    W_MAT[i,4] <- DT[i]

    #exit time of that patient from clinic
    W_MAT[i,5] <- sum(W_MAT[i,3],W_MAT[i,4])

    #updating Dr6 occupancy timestamp if this patient is in consultation with
doctor
    if(W_MAT[i,2] == 6){
      W_MAT[i,6] <- W_MAT[i,5]
    }else{
     W_MAT[i,6] <- W_MAT[i-1,6]
    }

    #updating Dr7 occupancy timestamp if this patient is in consultation with
doctor
    if(W_MAT[i,2] == 7){
      W_MAT[i,7] <- W_MAT[i,5]
    }else{
      W_MAT[i,7] <- W_MAT[i-1,7]
    }

    #updating Dr8 occupancy timestamp if this patient is in consultation with
doctor
    if(W_MAT[i,2] == 8){
      W_MAT[i,8] <- W_MAT[i,5]
    }else{
      W_MAT[i,8] <- W_MAT[i-1,8]
    }

    #determining if this patient had to wait or not
    W_MAT[i,9] <- (W_MAT[i,"Arr Time"] < W_MAT[i,"Starts"]) * 1

    #determining wait time if this patient had to wait
    W_MAT[i,10] <- (W_MAT[i,"Starts"]-W_MAT[i,"Arr Time"]) * W_MAT[i,9]

  }

  #in following manner, we will save each of this W_MAT matrix result of each
simulation in a list, which if required can be checked later:->
```

```
  All_W_MAT[[j]] <- W_MAT

  ##########
  # NOTE: just repeating same code as used and explained above in Q.1.(a), so
no need to explain all lines of code again
  ##########

  A2[j] <- sum(W_MAT[,"W"])


  A3[j] <- sum(W_MAT[,"WT"])/ A


  A4[j] <- max(max(W_MAT[A,c("Dr6","Dr7","Dr8")]),420)

}

# in case required, we can use following list to check the results matrix of
a particular simulation, for eg. to see 291th simulation matrix we can use
following code line:->
#All_W_MAT[[291]]

# in case required, we can use following vectors to check the summary of each
questions asked in Q.1.(a) for all of 1000 simulations
#A1
#A2
#A3
#A4

#now, as requested in the question Q.1.(b), we can estimate the median for
each of the above summaries vector A1 to A4 in the following manner:->
B1 <- median(A1)
B1

## [1] 42

B2 <- median(A2)
B2

## [1] 5

B3 <- median(A3)
B3

## [1] 0.4640054

B4 <- median(A4)
B4

## [1] 426.1528
```

```
#we can also convert above answer of B4 from minutes to clock time in
following manner, please note that we will be rounding of a fraction of
minute to higher side by applying ceiling() function:->
B4TH <- cat(floor((B4/60)-3),":",ceiling(B4%%60),"p.m.")

## 4 : 7 p.m.
```

so, as seen from running above code, by considering seed values of 1 to 1000 arbitrarily, we are getting following answers by simulating the process 1000 times and estimating the median for each of the summary questions asked in Q.1.(a):->

(i) How many patients came to the office? (median of 1000 simulations) Ans. 42

(ii) How many had to wait for a doctor? (median of 1000 simulations) Ans. 5

(iii) What was their average wait? (median of 1000 simulations) Ans. (0.4640054) minutes

(iv) When did the office close? (median of 1000 simulations) Ans. 4:07 p.m. (as mentioned in the code, we have rounded of 4:06.1528 p.m. to 4:07 p.m.)