project1 Q.3

Doris Kogan & Dov Tuch

4/25/2022

Question 3.1:

```r
# pre-process functions: (to use in all next functions)
# arrival time of pet:
time_pet = function(rate) {
  vec = numeric(1000)
  i = 1
  vec[1] = rexp(1, rate = rate)
  # build the vector of arrival time:
  while  (vec[i] < 720) {
    i = i+ 1
    vec[i] = rexp(1, rate = rate) + vec[i-1]
  }
  # return only values less then 720:
  return(vec[-i])
}

# the function of the duration of each treatment: gets the pet attribute and gives 
the duration:
tipul_duration <- function(pet_vec, index){
  name = names(pet_vec)[index]
  if ( name == "dog"){
    serv_time = rexp(1, rate = 3)
  }
  else{
    serv_time = rexp(1, rate = 5)
  }
  return(serv_time)
}

# the function of the duration of each treatment: gets the pets name and gives the 
duration: (use for 1 intor)
tipul_duration_pet <- function(name){
  kind = name
  if ( kind == "dog"){
    serv_time = rexp(1, rate = 3)
  }
  else{
    serv_time = rexp(1, rate = 5)
  }
  return(serv_time)
}

# the function of the payment of each treatment: gets the pet name and gives the pa
yment:
tipul_peyment_pet <-  function(name){

  kind = name
  if ( kind == "dog"){
    return(1)
  }
  else{
    return(3)
  }

}
```

```
# calculate the average line, using intervals and line size:
avg_tor_hat <- function(interval_time, tor_size){
  result = (sum(interval_time*tor_size))/720
  return(result)
}
```

```
#function starts here:
interval <-  function(){

  time_dog = time_pet(3)
  time_cat <- time_pet(1.5)

  # adding "pet name" attribute :

  names(time_dog) <-  rep("dog", length(time_dog))
  names(time_cat) <-  rep("cat", length(time_cat))

  # time of arrival of all costumers
  arrival  = sort(c(time_dog, time_cat))
  n = length(arrival)

  # starting the interval:
  int = numeric(5*n)
  int[1] = 0
  int[2] = arrival[1]
  i = 3

  # the line:
  tor = numeric(5*n)
  tor[1] = 0

  # end of treatment:
  sof_tipul = numeric(5*n)
  sof_tipul[1] = arrival[1] + tipul_duration(arrival, index = 1)

  # payment in bals:
  payment = numeric(5*n)
  payment[1] = tipul_peyment_pet(names(arrival[1]))
  # counter of dog rejection:
  dog_reject_count = 0

  tipul_num = 1
  arrival_num = 2
  tor_indx = 1
  # first_costumer_in_line = cat or dog names of next arrival when tor == 0'
  # only saved when arrival_time > sof_tipul
  while(int[i] < 720 & arrival_num < n - 1 ){
    # fail safe - just in case line gets values under 0:
    if (tor[tor_indx] < 0){
      tor[tor_indx] = 0
    }
    # case when tor = 0
    else if (tor[tor_indx] == 0){
      # reset the 1st in line
      rishon_intor = c()
      if (sof_tipul[tipul_num] <= arrival[arrival_num]){
        # the line didn't change start next tipul
        tipul_num = tipul_num + 1
        # The next sof_tipul is the time the pet arrived and it's tipul duration
        int[i] = arrival[arrival_num]
        sof_tipul[tipul_num] = arrival[arrival_num] + tipul_duration(arrival, index
= arrival_num)
        arrival_num = arrival_num+ 1
        i = i + 1
      }
      else{
        tor_indx = tor_indx + 1
        tor[tor_indx] = tor[tor_indx - 1] + 1
        int[i] = arrival[arrival_num]
        # heres the change, defining who is the 1st customer cat or dog
        rishon_intor = names(arrival[arrival_num])
        i = i + 1
        arrival_num = arrival_num + 1
      }
    }

    # case when line num betwwen 1-9
    else if (tor[tor_indx] > 0 && tor[tor_indx] < 10){
      if (sof_tipul[tipul_num] < arrival[arrival_num]){
        # the line decrease by 1 ,start next treatment:
        tor_indx = tor_indx + 1
        payment[tipul_num] = tipul_peyment_pet(rishon_intor)
        tor[tor_indx] = tor[tor_indx - 1] - 1
        tipul_num = tipul_num + 1
        int[i] = sof_tipul[tipul_num]
        # next treatment is imiddently after and the time it takes to service the f
irst in line
        sof_tipul[tipul_num] = sof_tipul[tipul_num - 1] + tipul_duration_pet(rishon
_intor)
        rishon_intor = "dog"
        i = i +1
      }

      else{
        if (names(arrival[arrival_num]) == "dog"){
          tor_indx = tor_indx + 1
          tor[tor_indx] = tor[tor_indx - 1] + 1
          arrival_num = arrival_num + 1
          int[i] = arrival[arrival_num]
          i = i +1
        }
        # a cat has arrived only update to the next costumer
        else{
          arrival_num = arrival_num + 1
        }
      }

    }

    #case when tor = 10 (is full)
    else{
      if (sof_tipul[tipul_num] <= arrival[arrival_num]){
        tor_indx = tor_indx + 1
        tor[tor_indx] = tor[tor_indx - 1] - 1
        int[i] = sof_tipul[tipul_num]
        payment[tipul_num] = tipul_peyment_pet(rishon_intor)
        tipul_num = tipul_num + 1
        # next tipul is imiddently after and the time it takes to service the first in line
        sof_tipul[tipul_num] = sof_tipul[tipul_num - 1] + tipul_duration_pet(rishon_intor)
        rishon_intor = "dog"
        i = i +1
      }
      # a customer arrived and tor is full
      else{
        if(names(arrival[arrival_num]) == "dog"){
          dog_reject_count = dog_reject_count + 1
          arrival_num = arrival_num + 1
        }
        else{
          arrival_num = arrival_num + 1
        }

      }
    }
  }

  time_int = numeric(i - 1)
  for (j in (2:i)){
    time_int[j-1] = int[j] - int[j-1]
  }

  avg_estimate = avg_tor_hat(time_int, tor[1:(i-1 )])

  # return the avg tor
  profit = sum(payment) - 0.1*dog_reject_count
  dog_num = sum(payment == 1)
  cat_num = sum(payment == 3)
  cat_reject_count = length(time_cat) - cat_num
  answer_list = list("profit" = profit, "dog_num" = dog_num,"dog_reject_count" = dog_reject_count,"cat_num" =  cat_num, "cat_reject_count" = cat_reject_count , "est_avg_tor" = avg_estimate )

  return(answer_list)
}

sim_3a_profit = mean(replicate(100,interval()$profit))
sim_3a_avg = mean(replicate(100,interval()$est_avg_tor))
sim_3a_dog_num = mean(replicate(100,interval()$dog_num))
sim_3a_cat_num = mean(replicate(100,interval()$cat_num))
sim_3a_reject_count = mean(replicate(100,interval()$dog_reject_count))
sim_3a_reject_cat_count = mean(replicate(100,interval()$cat_reject_count))

paste("The expected value of profit:" , round(sim_3a_profit,3))
```

```
## [1] "The expected value of profit: 2069.373"
```

```
paste("The expected value of avg tor:" , round(sim_3a_avg,3))
```

```
## [1] "The expected value of avg tor: 5.788"
```

```
paste("The expected value of dogs served:" , round(sim_3a_dog_num ,3))
```

```
## [1] "The expected value of dogs served: 1859.49"
```

```
paste("The expected value of cats served:" , round(sim_3a_cat_num,3))
```

```
## [1] "The expected value of cats served: 74.92"
```

```
paste("The expected value of dog's rejected:" , round(sim_3a_reject_count,3))
```

```
## [1] "The expected value of dog's rejected: 190.06"
```

```
paste("The expected value of cat's rejected:" , round(sim_3a_reject_cat_count,3))
```

```
## [1] "The expected value of cat's rejected: 1002.91"
```

Question 3.2.a:

החסרון באפשרות זו הוא שמספר החתולים שמקבלים שירות קטן, ולכן הם סובלים מהצעה זו.

בגלל שהתור מכיל 20 מקומות המתנה, אורך התור הממוצע גדל, ולכן יותר ויותר חתולים יגיעו לתור כאשר אינו ריק

```
interval_2a <-  function(){

  time_dog = time_pet(3)
  time_cat <- time_pet(1.5)

  # adding "pet name" attribute

  names(time_dog) <-  rep("dog", length(time_dog))
  names(time_cat) <-  rep("cat", length(time_cat))

  # time of arrival of all costumers
  arrival  = sort(c(time_dog, time_cat))
  n = length(arrival)


  int = numeric(5*n)
  int[1] = 0
  int[2] = arrival[1]
  i = 3

  tor = numeric(5*n)
  tor[1] = 0


  sof_tipul = numeric(5*n)
  sof_tipul[1] = arrival[1] + tipul_duration(arrival, index = 1)

  # payment in bals
  payment = numeric(5*n)
  payment[1] = tipul_peyment_pet(names(arrival[1]))
  # counter of dog rejection
  dog_reject_count = 0


  tipul_num = 1
  arrival_num = 2
  tor_indx = 1
  # first_costumer_in_line = cat or dog names of next arrival when tor == 0'
  # only saved when arrival_time > sof_tipul
  while(int[i] < 720 & arrival_num < n - 1 ){
    # fail safe
    if (tor[tor_indx] < 0){
      tor[tor_indx] = 0
    }
    # case when line = 0
    else if (tor[tor_indx] == 0){
      # reset the 1st in line
      rishon_intor = c()
      if (sof_tipul[tipul_num] <= arrival[arrival_num]){
        # the tor didn't change start next treatment
        tipul_num = tipul_num + 1
        # The next sof_tipul is the time the pet arrived and it's treatment duration
        int[i] = arrival[arrival_num]
        sof_tipul[tipul_num] = arrival[arrival_num] + tipul_duration(arrival, index
= arrival_num)
        arrival_num = arrival_num+ 1
        i = i + 1
      }
      else{
        tor_indx = tor_indx + 1
        tor[tor_indx] = tor[tor_indx - 1] + 1
        int[i] = arrival[arrival_num]
        # heres the change, defining who is the 1st customer cat or dog
        rishon_intor = names(arrival[arrival_num])
        i = i + 1
        arrival_num = arrival_num + 1
      }
    }

    # case when line num betwwen 1-19
    else if (tor[tor_indx] > 0 && tor[tor_indx] < 19){
      if (sof_tipul[tipul_num] < arrival[arrival_num]){
        # the line sevrese by 1 ,start next treatment
        tor_indx = tor_indx + 1
        payment[tipul_num] = tipul_peyment_pet(rishon_intor)
        tor[tor_indx] = tor[tor_indx - 1] - 1
        tipul_num = tipul_num + 1
        int[i] = sof_tipul[tipul_num]
        # next treatment is imiddently after and the time it takes to service the f
irst in line
        sof_tipul[tipul_num] = sof_tipul[tipul_num - 1] + tipul_duration_pet(rishon
_intor)
        rishon_intor = "dog"
        i = i +1
      }

      else{
        if (names(arrival[arrival_num]) == "dog"){
          tor_indx = tor_indx + 1
          tor[tor_indx] = tor[tor_indx - 1] + 1
          arrival_num = arrival_num + 1
          int[i] = arrival[arrival_num]
          i = i +1
        }
        # a cat has arrived only update to the next costumer
        else{
          arrival_num = arrival_num + 1
        }
      }

    }

    #case when tor = 20 (is full)
    else{
      if (sof_tipul[tipul_num] <= arrival[arrival_num]){
        tor_indx = tor_indx + 1
        tor[tor_indx] = tor[tor_indx - 1] - 1
        int[i] = sof_tipul[tipul_num]
        payment[tipul_num] = tipul_peyment_pet(rishon_intor)
        tipul_num = tipul_num + 1
        # next tipul is imiddently after and the time it takes to service the rishon intor
        sof_tipul[tipul_num] = sof_tipul[tipul_num - 1] + tipul_duration_pet(rishon_intor)
        rishon_intor = "dog"
        i = i +1
      }
      # a customer arrived and tor is full
      else{
        if(names(arrival[arrival_num]) == "dog"){
          dog_reject_count = dog_reject_count + 1
          arrival_num = arrival_num + 1
        }
        else{
          arrival_num = arrival_num + 1
        }

      }
    }
  }

  time_int = numeric(i - 1)
  for (j in (2:i)){
    time_int[j-1] = int[j] - int[j-1]
  }

  avg_estimate = avg_tor_hat(time_int, tor[1:(i-1 )])

  # return the avg line
  profit = sum(payment) - 0.1*dog_reject_count
  dog_num = sum(payment == 1)
  cat_num = sum(payment == 3)
  cat_rejected = length(time_cat) - cat_num
  answer_list = list("profit" = profit, "dog_num" = dog_num,"dog_reject_count" = dog_reject_count,"cat_num" =  cat_num,
                     "est_avg_tor" = avg_estimate, "cat_rejected" = cat_rejected )

  return(answer_list)

}

sim_32a_profit = mean(replicate(100,interval_2a()$profit))
sim_32a_avg = mean(replicate(100,interval_2a()$est_avg_tor))
sim_32a_dog_num = mean(replicate(100,interval_2a()$dog_num))
sim_32a_cat_num = mean(replicate(100,interval_2a()$cat_num))
sim_32a_reject_count = mean(replicate(100,interval_2a()$dog_reject_count))
sim_32b_reject_cat = mean(replicate(100,interval_2a()$cat_rejected))

paste("The expected value of profit:" , round(sim_32a_profit,3))
```

```
## [1] "The expected value of profit: 2089.557"
```

```
paste("The expected value of avg tor:" , round(sim_32a_avg,3))
```

```
## [1] "The expected value of avg tor: 8.031"
```

```
paste("The expected value of dogs served:" , round(sim_32a_dog_num ,3))
```

```
## [1] "The expected value of dogs served: 1981.21"
```

```
paste("The expected value of cats served:" , round(sim_32a_cat_num,3))
```

```
## [1] "The expected value of cats served: 41.44"
```

```
paste("The expected value of dogs's rejected:" , round(sim_32a_reject_count,3))
```

```
## [1] "The expected value of dogs's rejected: 112.55"
```

```
paste("The expected value of cat's rejected:" , round(sim_32b_reject_cat,3))
```

```
## [1] "The expected value of cat's rejected: 1033.24"
```

Question 3.2.b:

```
interval_2b = function(){
  tipul_duration2 <- function(pet_vec, index){
    name = names(pet_vec)[index]
    if ( name == "dog"){
      serv_time = rexp(1, rate = 3.3)
    }
    else{
      serv_time = rexp(1, rate = 5.5)
    }
    return(serv_time)
  }

  tipul_duration_pet2 <- function(name){
    kind = name
    if ( kind == "dog"){
      serv_time = rexp(1, rate = 3.3)
    }
    else{
      serv_time = rexp(1, rate = 5.5)
    }
    return(serv_time)
  }

  time_dog = time_pet(3)
  time_cat <- time_pet(1.5)

  # adding "pet name" attribute

  names(time_dog) <-  rep("dog", length(time_dog))
  names(time_cat) <-  rep("cat", length(time_cat))

  # time of arrival of all costumers
  arrival  = sort(c(time_dog, time_cat))
  n = length(arrival)


  int = numeric(5*n)
  int[1] = 0
  int[2] = arrival[1]
  i = 3

  tor = numeric(5*n)
  tor[1] = 0


  sof_tipul = numeric(5*n)
  sof_tipul[1] = arrival[1] + tipul_duration2(arrival, index = 1)

  # payment in bals
  payment = numeric(5*n)
  payment[1] = tipul_peyment_pet(names(arrival[1]))
  # counter of dog rejection
  dog_reject_count = 0
  tipul_num = 1
  arrival_num = 2
  tor_indx = 1
  # first_costumer_in_line = cat or dog names of next arrival when tor == 0'
  # only saved when arrival_time > sof_tipul
  while(int[i] < 720 & arrival_num < n - 1 ){
    # fail safe
    if (tor[tor_indx] < 0){
      tor[tor_indx] = 0
    }
    # case when line = 0
    else if (tor[tor_indx] == 0){
      # reset the 1st in line
      rishon_intor = c()
      if (sof_tipul[tipul_num] <= arrival[arrival_num]){
        # the line didn't change start next treatment
        tipul_num = tipul_num + 1
        # The next sof_tipul is the time the pet arrived and it's treatment duratio
n
        int[i] = arrival[arrival_num]
        sof_tipul[tipul_num] = arrival[arrival_num] + tipul_duration2(arrival, inde
x = arrival_num)
        arrival_num = arrival_num+ 1
        i = i + 1
      }
      else{
        tor_indx = tor_indx + 1
        tor[tor_indx] = tor[tor_indx - 1] + 1
        int[i] = arrival[arrival_num]
        # heres the change, defining who is the 1st customer cat or dog
        rishon_intor = names(arrival[arrival_num])
        i = i + 1
        arrival_num = arrival_num + 1
      }
    }

    # case when line num betwwen 1-9
    else if (tor[tor_indx] > 0 && tor[tor_indx] < 10){
      if (sof_tipul[tipul_num] < arrival[arrival_num]){
        # the tor yored 1 ,start next tipul
        tor_indx = tor_indx + 1
        payment[tipul_num] = tipul_peyment_pet(rishon_intor)
        tor[tor_indx] = tor[tor_indx - 1] - 1
        tipul_num = tipul_num + 1
        int[i] = sof_tipul[tipul_num]
        # next tipul is imiddently after and the time it takes to service the risho
n intor
        sof_tipul[tipul_num] = sof_tipul[tipul_num - 1] + tipul_duration_pet2(risho
n_intor)
        rishon_intor = "dog"
        i = i +1
      }
      else{
        if (names(arrival[arrival_num]) == "dog"){
          tor_indx = tor_indx + 1
          tor[tor_indx] = tor[tor_indx - 1] + 1
          arrival_num = arrival_num + 1
          int[i] = arrival[arrival_num]
          i = i +1
        }
        # a cat has arrived' only update to the next costumer
        else{
          arrival_num = arrival_num + 1
        }
      }

    }

    #case when tor = 10 (is full)
    else{
      if (sof_tipul[tipul_num] <= arrival[arrival_num]){
        tor_indx = tor_indx + 1

        tor[tor_indx] = tor[tor_indx - 1] - 1
        int[i] = sof_tipul[tipul_num]
        payment[tipul_num] = tipul_peyment_pet(rishon_intor)
        tipul_num = tipul_num + 1
        # next tipul is imiddently after and the time it takes to service the risho
n intor
        sof_tipul[tipul_num] = sof_tipul[tipul_num - 1] + tipul_duration_pet2(risho
n_intor)
        rishon_intor = "dog"
        i = i +1
      }
      # a customer arrived and tor is full
      else{
        if(names(arrival[arrival_num]) == "dog"){
          dog_reject_count = dog_reject_count + 1
          arrival_num = arrival_num + 1
        }
        else{
          arrival_num = arrival_num + 1
        }

      }
    }
  }

  time_int = numeric(i - 1)
  for (j in (2:i)){
    time_int[j-1] = int[j] - int[j-1]
  }

  avg_estimate = avg_tor_hat(time_int, tor[1:(i-1 )])

  # return the avg tor
  profit = sum(payment) - 0.1*dog_reject_count
  dog_num = sum(payment == 1)
  cat_num = sum(payment == 3)
  cat_rejected = length(time_cat) - cat_num
  answer_list = list("profit" = profit, "dog_num" = dog_num,"dog_reject_count" = do
g_reject_count,"cat_num" =  cat_num, "est_avg_tor" = avg_estimate, "cat_rejected" =
cat_rejected )

  return(answer_list)



}

sim_32b_profit = mean(replicate(100,interval_2b()$profit))
sim_32b_avg = mean(replicate(100,interval_2b()$est_avg_tor))
sim_32b_dog_num = mean(replicate(100,interval_2b()$dog_num))
sim_32b_cat_num = mean(replicate(100,interval_2b()$cat_num))
sim_32b_reject_count = mean(replicate(100,interval_2b()$dog_reject_count))
sim_32b_reject_cat = mean(replicate(100,interval_2b()$cat_rejected))

paste("The expected value of profit:" , round(sim_32b_profit,3))
```

```
## [1] "The expected value of profit: 2187.848"
```

```
paste("The expected value of avg tor:" , round(sim_32b_avg,3))
```

```
## [1] "The expected value of avg tor: 5.224"
```

```
paste("The expected value of dogs served:" , round(sim_32b_dog_num ,3))
```

```
## [1] "The expected value of dogs served: 1868.61"
```

```
paste("The expected value of cats served:" , round(sim_32b_cat_num,3))
```

```
## [1] "The expected value of cats served: 108.63"
```

```
paste("The expected value of dog's rejected:" , round(sim_32b_reject_count,3))
```

```
## [1] "The expected value of dog's rejected: 104.15"
```

```
paste("The expected value of cat's rejected:" , round(sim_32b_reject_cat,3))
```

```
## [1] "The expected value of cat's rejected: 966.88"
```

לא הספקנו לסיים את סעיף ג :(

מבין סעיפים א וב האפשרות העדיפה לחברה הינה אפשרות ב, שכן הרווח בה גדול יותר.

זאת משום שזמן השירות מתקצר ולכן ניתן לטפל במספר גדול יותר של לקוחות.