Faculty Of Computing Informatics

TMA1301 (Computational Methods)

Trimester 2, 2021/2022

Queueing System

Tutorial Section: TT5V

Group Members:

No	Name	Id No
1.	Ayat Abdulaziz Gaber Al-Khulaqi	1191202335
2.	Iven Low Zi Yin	1191202539
3.	Mohammad Harez Bin Hafez	1191202413

Table of Contents

Queueing System :

1. Random Number Generators :-

To use the queueing system, users must first type in the number of patients. The number of patients is important since the system would need to generate the random number. Then, the user can choose two choices of random number generator.

There are two ways of creating Random number generators, the first one is Linear Congruential Generators (LCG). The formula for this LCG is X= (aX c)(m), this was the base formula. In order to always generate random numbers until the number of patients is equal to the number of random numbers for every one of them, all of the variables will generate a random number of their own.

The next type of random number generator is a Simple Rand Function. It is a built-in function to generate random numbers. All of the variables will be generated by random numbers using the function (rand).

First, the generator will generate the random number as many as the user wants depending on how many patients that the user enters. Meaning that if the number of patients is 10, then the generator will generate 10 random numbers for inter-arrival time but we have assigned the first inter-arrival time of the first patient is zero because the starting number will always be number zero. After calculating the inter-arrival time, then the generator will continue to calculate for service time using the same method. Unlike the inter-arrival time calculation, service time will continue to produce random numbers until the counter matches the number of patients and the first random number can be any value.

After the calculation of generating random numbers using the user choices of random number generator either the Linear Congruential Generators (LCG) or the Simple Rand Function, the system will display the number that have been generated for inter-arrival time and service time.

Flow-Charts :

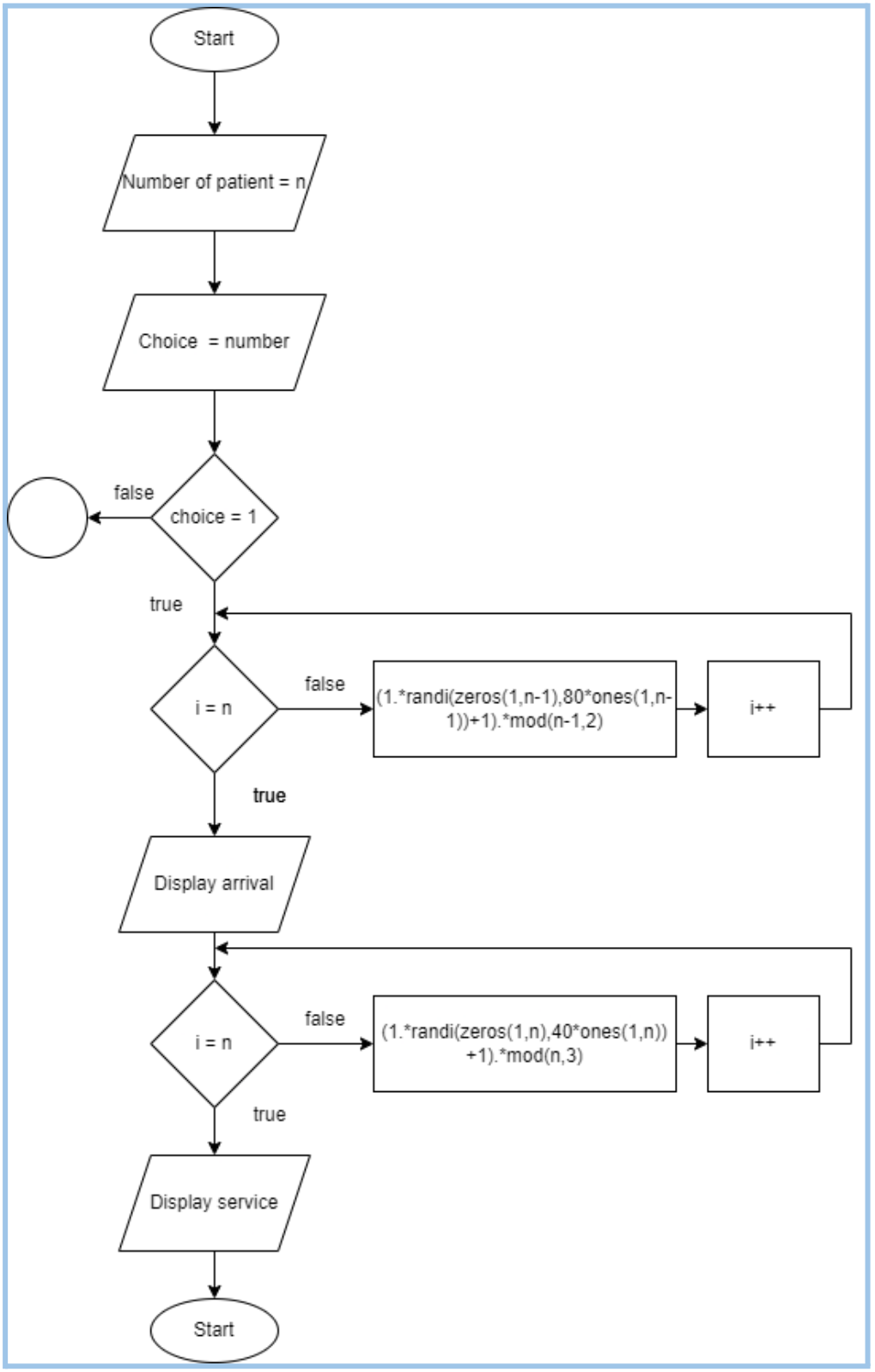

Chart 1.1 Random number generator choice 1 is Linear Congruential Generators

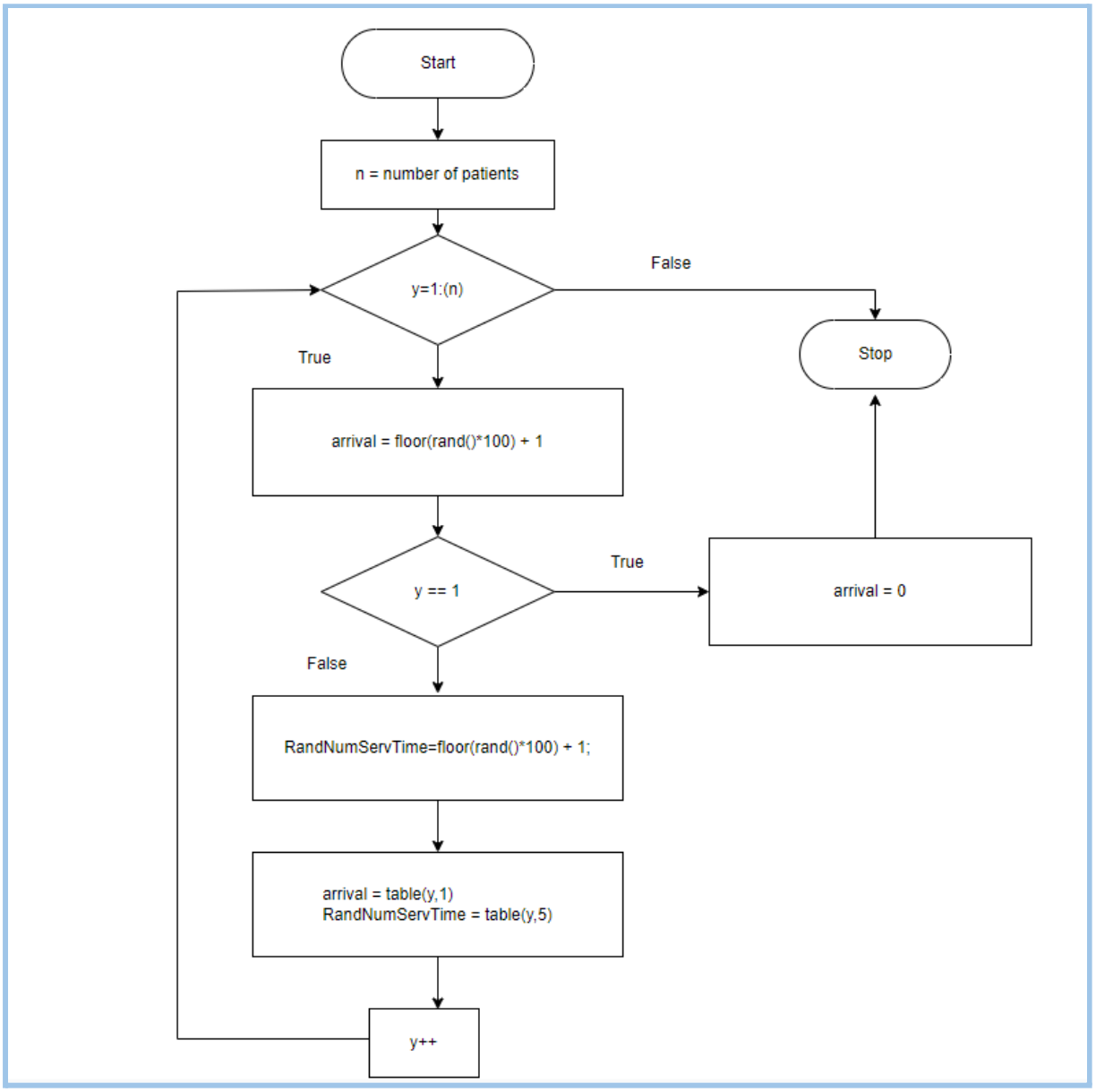

Chart 1.2 Random number generator choice 2 is Simple Random Function

Formulas Used in the System :

```
a=(1.*randi(zeros(1,n-1),80*ones(1,n-1))+1).*mod(n-1,2);
arrival = randi(zeros(1,n),zeros(1,n));
arrival(1:1,2:n) = a;
fprintf('Random Number For Inter Arrival Time:    '); disp(a);
service=(1.*randi(zeros(1,n),40*ones(1,n))+1).*mod(n,3);
fprintf('Random Number For Service Time      : ');  disp(service);
```

Code 1.1 Random number generator choice 1 is Linear Congruential Generators Formula

As you can see in (*Code 1.1*) the formula for the first choice of random number generator is Linear Congruential Generators. The actual formula is X= (aX c)(m) in which the variables of X are the starting values, m is the modulus, c is the increment and a is the multiplier. The formula in the code uses the (randi) function to generate for all the variables.

```
arrival =floor(rand()*100) + 1;

if(y == 1)
    RNInterArrival=0;
    InterArrival = 0;
    ArrivalTime = 0;
    arrival=0;
end
```

Code 1.2 Random number generator choice 2 is Simple Random Function

For the formula of choice 2 in (*Code 1.2*), the floor function is used to round each element of the variable to the nearest integer.

All of the numbers for each patient will be auto generated randomly and assigned to the variable for each patient in the for loop . For the first patient the inter-arrival time will need to be zero because the first inter-arrival time would be zero. For the service time the number of patients (n) does not need to have extra conditions as the service time can be any number. The formula for both Linear Congruential Generators (LCG) and Exponential Distribution have been configured to produce random numbers with ranges between 1 to 100.

Screenshots :

```
--> start
Welcome to Hostpital self-service registration!

Please enter number of patients : 10


_____________________________________________________________________
           Choose The Random Number Generator List
_____________________________________________________________________

1. Choice 1
2. Choice 2
_____________________________________________________________________

Please enter random number choice :1


Random Number For Inter Arrival Time:   0  5  8 30 20 17 26  1 14 35

Random Number For Service Time      :  18 17  9 26 17 15 19 32 27 30
```

Figure 1.1 Random number generator choice 1 is Linear Congruential Generators

```
--> start
Welcome to Hostpital self-service registration!

Please enter number of patients : 10


_____________________________________________________________________
           Choose The Random Number Generator List
_____________________________________________________________________

1. Choice 1
2. Choice 2
_____________________________________________________________________

Please enter random number choice :2


Random Number For Inter Arrival Time:   0  5  5 38 38 15  1 54  7 14

Random Number For Service Time      :  19 19  8 93 33 68 43 10  8 17
```

Figure 1.2 Random number generator choice 2 is Simple Rand Function

2. Round Robin : -

For this queue system was simulated according to the round robin waiting line rule where the first patient is sent to Kiosk 1 and the second to Kiosk#2, the third to Kiosk#1, the fourth to Kiosk#2, and so on. After the user typed the number of patients and chose choice 1 as shown in (*Figure 1.1*).

Then the system will display the Inter-Arrival Time Table and Service Time Tables for Kiosk 1 and Kiosk 2, as seen in (*Figure 2.1 - Figure 2.2*). The inter-arrival time and service time tables both of them are fixed values. To calculate the inter-arrival time for each patient, a for loop and inter-arrival time array were created. The for loop is looped based on the number of patients. To find the inter-arrival for each patient, an if-else statement was created inside the for loop as seen in (*Chart 2.1*). For example, if the patient RN inter-arrival time is 25 then the inter-arrival time is 2 and this number will be saved in an array.

To calculate the arrival time for each patient a for loop, three variables, arrival time and departure time arrays were created and the inter-arrival array was used. Also, the for loop is looped based on the number of patients. Inside the for loop there are two if-else statements one to calculate the arrival time and the other to calculate the departure time. The for loop also displays the arrival time and departure time as illustrated in (*Figure 2.3*).

For the Simulation Table Part 1, the number of patients, RN inter-arrival time, inter-arrival time, and arrival time were displayed as shown in (*Figure 2.4*). So, another for loop was created based on the number of the patients. The RN inter-arrival time, inter-arrival time and arrival time arrays were used in the for loop.

For the Simulation Table Part 2 (*Figure 2.5*), the patients were divided between two kiosk 1 and kiosk 2. As mentioned before that the first patient is sent to Kiosk 1 and the second to Kiosk#2, the third to Kiosk#1, and so on. A for loop was created and inside the for loop there is an if else statement which divides the patients between kiosk 1 and kiosk 2. So, if the patient number divided by two equals zero then the

patient is passed to kiosk 2 else it will be passed to kiosk 1. There are another two if-else statements inside the first one to calculate the service time for both kiosks, and the value is saved in kiosk 1 and kiosk 2 arrays. The array size for both kiosks depends on the number of patients. If the patient number is an odd number then the array size for kiosk 1 is the (number of patients plus one divided by two) and for kiosk 2 is (number of patients minus kiosk 1), as shown in (*Chart 2.2*). For example, if we have 51 patients then we will have 26 patients in kiosk 1 and 25 in kiosk 2.

For the service time begins, the value is when the patient arrives which uses the arrival time array to get the values. To calculate service time ends for each patient, the patient service time and service time begins are added together. To calculate the waiting time is based on the two kiosks, for example patient 4 they had to wait for 1 minute, because the waiting time = Time Service ends (patient3) - Time Service begins (patient4) and the value is saved in an array. The time spent is the service time of each patient spent as shown in (*Figure 2.5*).

Lastly for the Results Evaluation (*Figure 2.6*), from the waiting time array we calculate the total waiting time then divided by the number of patients to get the average waiting time. For the average time spent in the registration system is the total value of kiosk 1 and kiosk 2 then divided by the number of the patients. For the probability that a patient has to wait is the average waiting time. The average time of kiosk 1 is calculated by adding all kiosk 1 values then dividing it by the number of patients in kiosk 1, the same process goes for kiosk 2.

Flow-Charts :

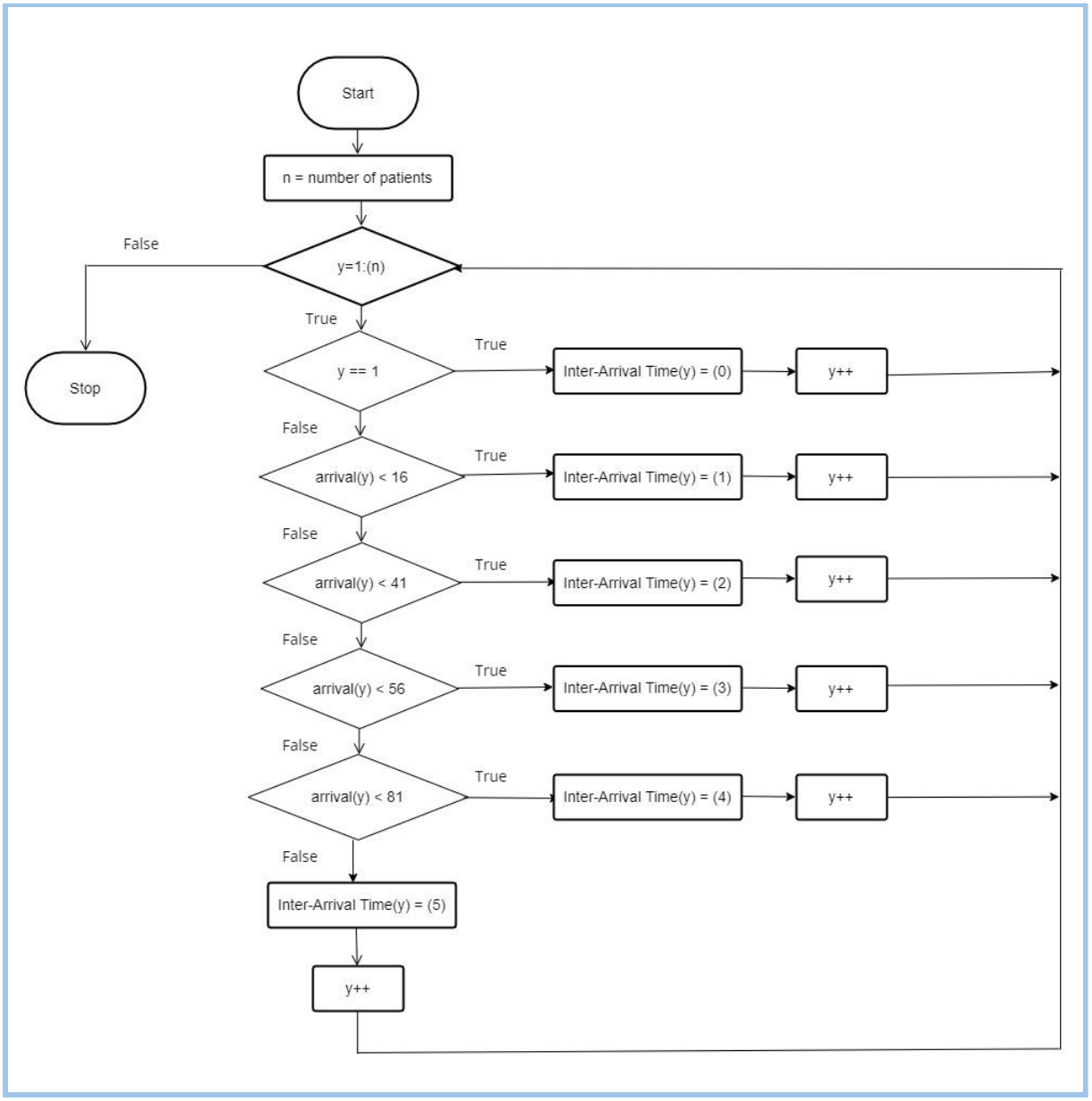

Chart 2.1 Inter-Arrival Time

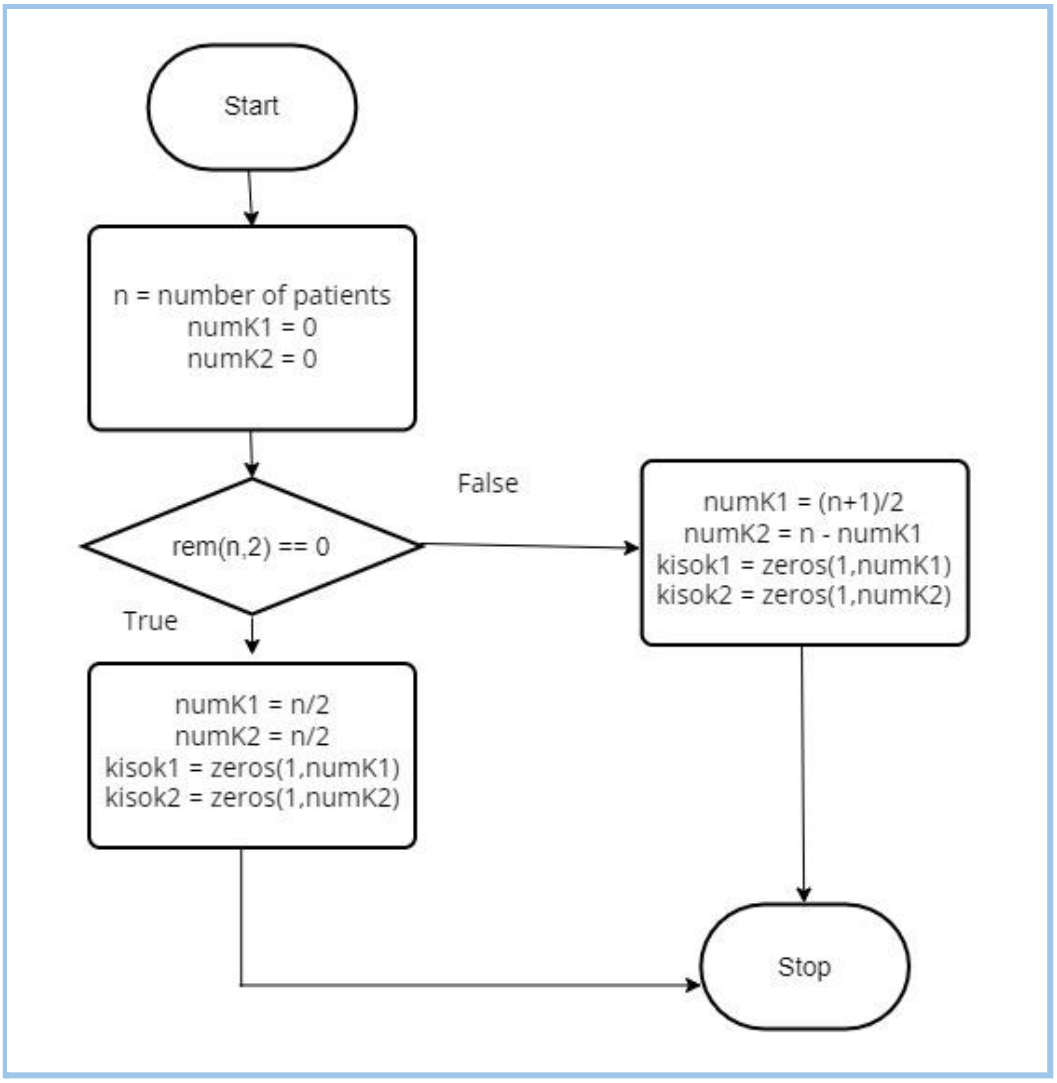

Chart 2.2 Array size for kiosk1 & kiosk2

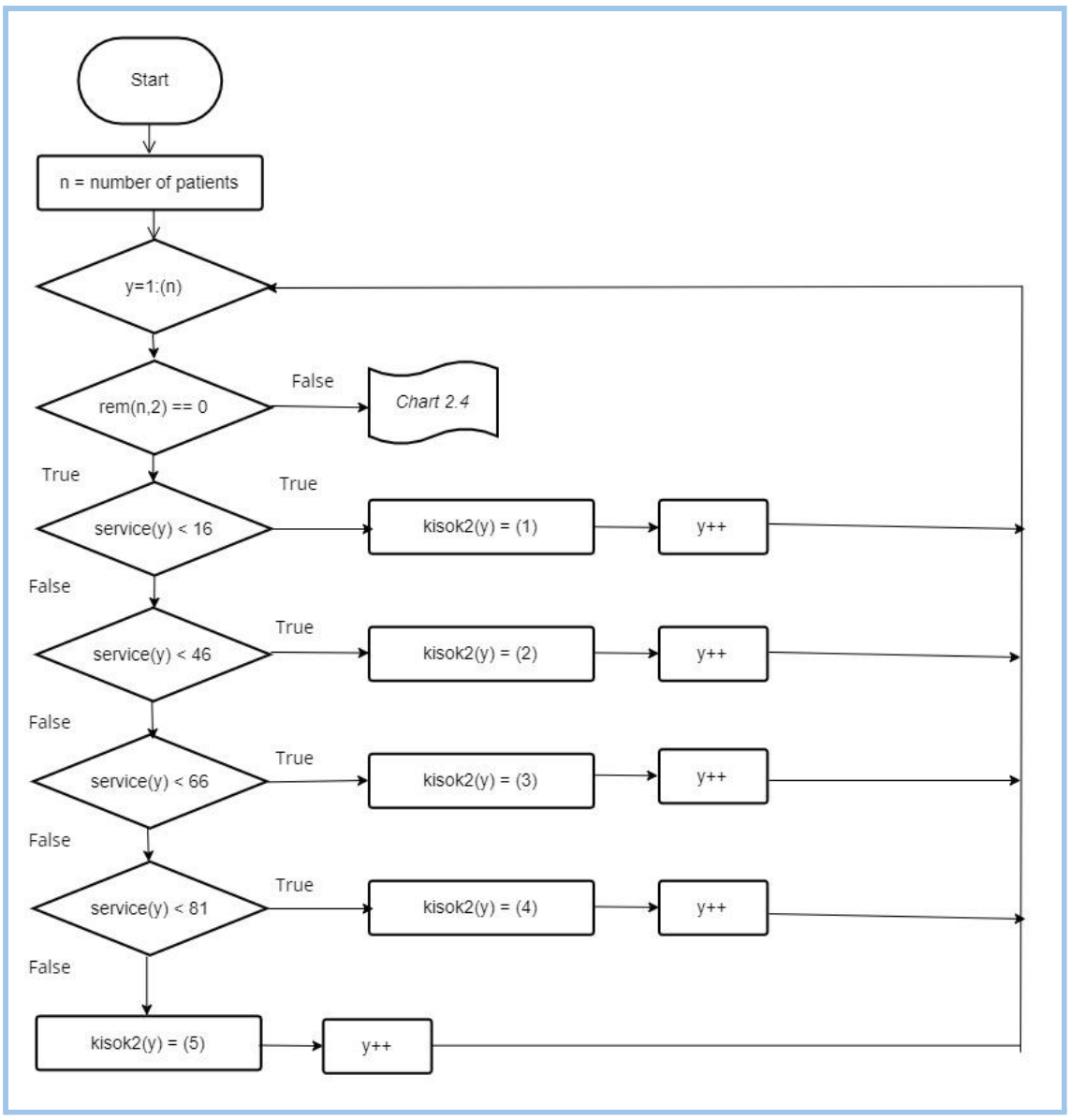

Chart 2.3 Kiosk 2 Service Time Chart

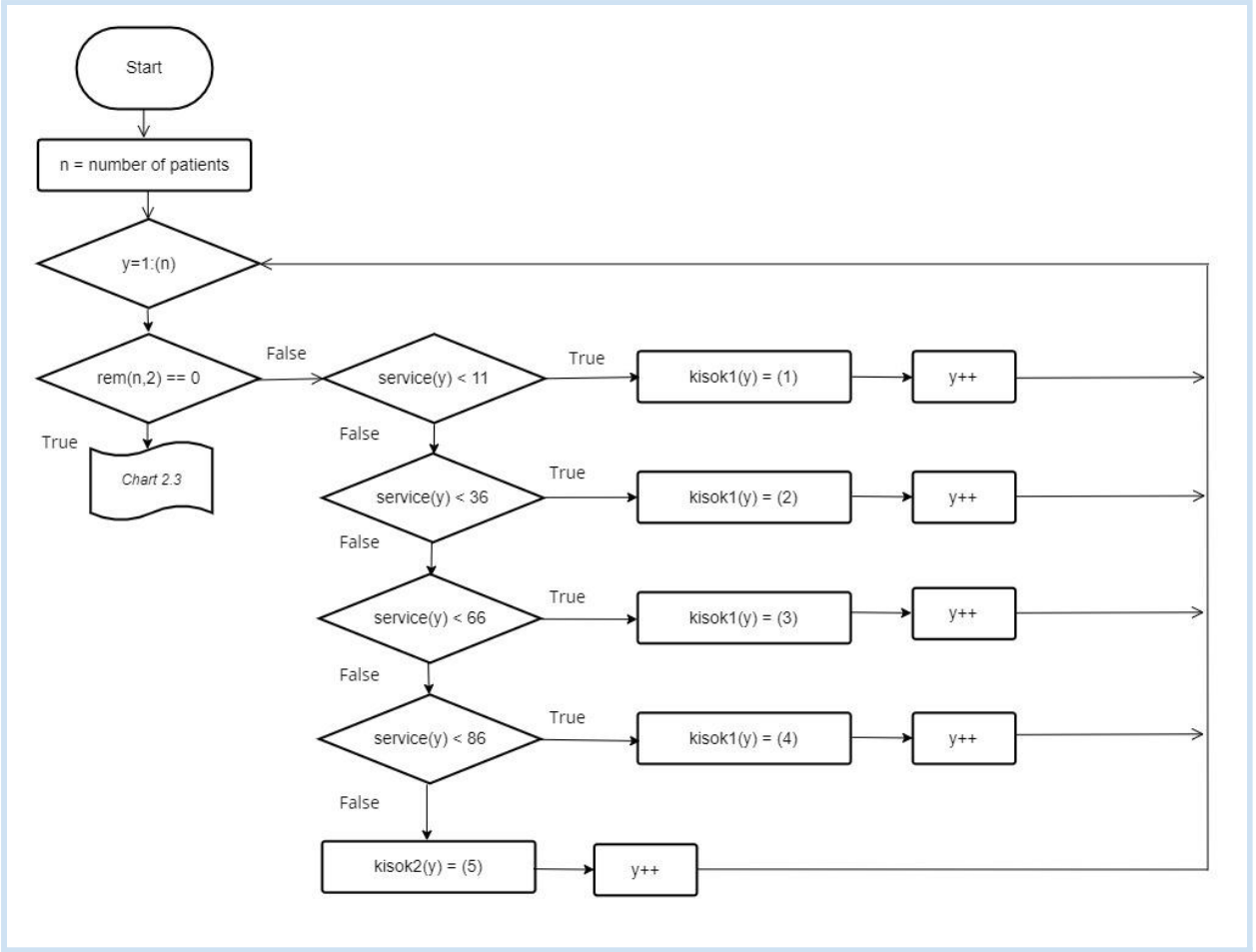

Chart 2.4 Kiosk 1 Service Time Chart

Formulas Used in the System :

The calculation for the arrival time and departure time, here is the coding :

```
    fprintf('\n_____________________________________
    fprintf('Arrival and Departure Details:
    fprintf('\n_____________________________________

    for y=1:(n)

    num = y+1;

    if (y == 1)
    AD = AT + InterArrivalTime(2);
    DepartTime(y) = (AD);
    else
    AD =  InterArrivalTime(y) + AT;
    DepartTime(y) = AD;
    end

    fprintf('\nPatient %1d ',y);
    fprintf('arrives at minute %1d',  AT );
    ArrivalTime(y) = AT;

    if(num > n)
        num = n-1;
    else
        AT = AT + InterArrivalTime(num);
    end

    fprintf('\nPatient %1d ',y);
    fprintf('departs at minute %1d',  AD );
    fprintf('\n\n');
end
```

Code 2.1 Arrival & Departure time Calculation

First of all, three variables were created (num, AT and AD), all of them were initialised to zero. To calculate the departure time (AD) for the first patient is equal to arrival time (AT) added to the second patient inter-arrival (InterArrivalTime). However, from the second to the last patient, the departure time (AD) is calculated by adding the inter-arrival time (InterArrivalTime) of the patient added to arrival time (AT). As you can see in (*Code 2.1),* the arrival time displayed first before the if -else statement. After displaying the first patient arrival time (AT), arrival time (AT) is equal to arrival time(AT) added to (InterArrivalTime) of the patient plus one (num). For the last patient (num) would be larger than the number of patients(n), so (num) is minus by one to calculate the last arrival time of the last patient.

```
numK1 = 0;
numK2 = 0;

% The array size for kiosks depends on the number of patients
if(rem(n,2) ==0)

    numK1 = n/2;
    numK2 = n/2;
    %kiosk 1 Array
    kisok1 = zeros(1,numK1);
    %kiosk 2 Array
    kisok2 = zeros(1,numK2);
else
    numK1 = (n+1)/2;
    numK2 = (n - numK1);
    kisok1 = zeros(1,numK1);
    %kiosk 2 Array
    kisok2 = zeros(1,numK2);
end
```

Code 2.2 Array size for kiosk1 & kiosk2

To find the array size of kiosk 1 and kiosk 2, first we find the remainder of the number of patients divided by two. If the remainder is zero, then the size of kiosk 1 and kiosk 2 is the number of patients divided by two. However, if the remainder is not zero then the array size for kiosk 1 (numk1) is the number of patients added one then divided by two. The array size for kiosk 2 (numk2) is the number of patients minus the array size for kiosk 1 (numk1).

```
for y=1:(n)

    fprintf('%5.0f      |',y);
    fprintf('%8.0f       |',service(y));

    if(rem(y,2) == 0)
    fprintf('%45.0f     |',kisok2(y));
    fprintf('%8.0f       |',ArrivalTime(y));
    TSE2 = kisok2(y) + ArrivalTime(y);
    fprintf('%8.0f       |',TSE2);
    TSEArr(y) = TSE2;


    else

    fprintf('%5.0f     |',kisok1(y));
    fprintf('%8.0f       |',ArrivalTime(y));
    TSE1 = kisok1(y) + ArrivalTime(y);
    fprintf('%8.0f       |',TSE1);
    TSEArr(y) = TSE1;

    end
```

Code 2.3 Time service end for kiosk 1& kiosk2

To calculate the time service ends, if the remainder of the number of patients divided by two is zero. Then the time service ends for kiosk 2 is kiosk 2 service time (kiosk2) added to the arrival time (ArrivalTime) of the patient. However, if the remainder is not zero, then the time service ends for kiosk 1 is kiosk 1 service time (kiosk1) added to the arrival time (ArrivalTime) of the patient. Lastly the values for time service ends for both kiosks are saved in an array (TSEArr).

```
if(y == 1)
    WT = 0;
    fprintf('%45.0f    |',WT);
    WTArr(y) = WT;
elseif(TSEArr(y-1) > ArrivalTime(y))

    WT = TSEArr(y-1) - ArrivalTime(y);
    WTArr(y) = WT;
    if(rem(y,2) == 0)
        fprintf('%5.0f    |',WT);
    else
    fprintf('%45.0f    |',WT);
    end
else
    WT = 0;
    WTArr(y) = WT;
    if(rem(y,2) == 0)
        fprintf('%5.0f    |',WT);
    else
    fprintf('%45.0f    |',WT);
    end
end
```

Code 2.3 Waiting Time Calculation

For the waiting time (WT), if it is the first patient then waiting time is zero. However, if it's not the first patient then if the service time end of the previous patient is bigger than the arrival time (ArrivalTime) of the patient. Then waiting time (WT) is service time at the end of the previous patient minus the arrival time (ArrivalTime). If the service time end of the previous patient is not greater than the arrival time (ArrivalTime) of the patient, then waiting time is zero. All waiting time values are saved in an array.

```
if(rem(y,2) == 0)
fprintf('%5.0f    ',kisok2(y));
ATSRSTotal = ATSRSTotal + kisok2(y);
Totalkisok2 = Totalkisok2 + kisok2(y);

else
   fprintf('%5.0f    ',kisok1(y));
   ATSRSTotal = ATSRSTotal + kisok1(y);
   Totalkisok1 = Totalkisok1 + kisok1(y);
end

TotalWT = TotalWT +  WTArr(y);

fprintf('\n');

end
```

```
fprintf('Average Waiting Time : ');
disp(TotalWT/n);
fprintf('Average Time Spent in the registration system : ');
disp(ATSRSTotal/n);
fprintf('Probability that a patient has to wait : ');
disp(TotalWT/n);
fprintf('Average Service Time of Kiosk 1 :');
disp(Totalkisok1/numK1);
fprintf('Average Service Time of Kiosk 2 :');
disp(Totalkisok2/numK2);
```

Code 2.4 Results Evaluation *Calculations*

To calculate the average waiting time, first of all we get the total waiting time then divide it by the total number of patients. To get the total time spent in the registration system a variable was created (ATSRSTotal), to calculate the total of both kiosk 1 and kiosk 2. For the average time spent in the registration system total time spent in the registration system divided by the total number of patients. To get the average service time of kiosk 1, first get the total of kiosk 1(Totalkiosk1) then divide it by the total number of patients, the same process goes for kiosk2.

Screenshots :

Inter-Arrival Time Table

Inter-Arrival Time	Probability	CDF	Range
1	0.15	0.15	1-15
2	0.40	0.40	16-40
3	0.55	0.55	41-55
4	0.00	0.80	56-80
5	1.00	1.00	81-100

Figure 2.1 Inter-Arrival Time Table

Service Time for kiosk 1 Table

Service Time for kiosk 1	Probability	CDF	Range
1	0.10	0.10	1-10
2	0.25	0.35	11-35
3	0.15	0.65	36-65
4	0.25	0.85	66-85
5	0.20	1.00	86-100

Service Time for kiosk 2 Table

Service Time for kiosk 2	Probability	CDF	Range
1	0.15	0.15	1-15
2	0.30	0.45	16-45
3	0.20	0.65	46-65
4	0.15	0.80	66-80
5	0.20	1.00	81-100

Figure 2.2 Service Time Table for Kiosk 1&2

```
Arrival and Departure Details:

Patient 1 arrives at minute 0
Patient 1 departs at minute 3

Patient 2 arrives at minute 3
Patient 2 departs at minute 6

Patient 3 arrives at minute 6
Patient 3 departs at minute 9

Patient 4 arrives at minute 7
Patient 4 departs at minute 8

Patient 5 arrives at minute 10
Patient 5 departs at minute 13

Patient 6 arrives at minute 13
Patient 6 departs at minute 16

Patient 7 arrives at minute 15
Patient 7 departs at minute 17

Patient 8 arrives at minute 17
Patient 8 departs at minute 19

Patient 9 arrives at minute 19
Patient 9 departs at minute 21

Patient 10 arrives at minute 21
Patient 10 departs at minute 23
```

Figure 2.3 Arrival and Departure Details

SIMULATION TABLE Part 1

Patient	RN for Inter-arrival Time	Inter-arrival Time	Arrival Time
1	0	0	0
2	43	3	3
3	55	3	6
4	2	1	7
5	49	3	10
6	46	3	13
7	35	2	15
8	22	2	17
9	25	2	19
10	31	2	21

Figure 2.4 Simulation Table Part 1

SIMULATION TABLE Part 2

		kiosk 1			kiosk 2				
Patient	RN For Service Time	Service Time	Time Service begins	Time Service ends	Service Time	Time Service begins	Time Service ends	Waiting time	Time spend
1	14	2	0	2				0	2
2	8				1	3	4	0	1
3	16	2	6	8				0	2
4	15				1	7	8	1	1
5	40	3	10	13				0	3
6	12				1	13	14	0	1
7	37	3	15	18				0	3
8	15				1	17	18	1	1
9	9	1	19	20				0	1
10	8				1	21	22	0	1

Figure 2.5 Simulation Table Part 2

```
Results Evaluation:

Average Waiting Time :     0.2000

Average Time Spent in the registration system :     1.6000

Probabilitythat a patient has to wait :     0.2000

Average Service Time of Kiosk 1 :    2.2000

Average Service Time of Kiosk 2 : 1

--> 
```

Figure 2.6 Results Evaluation

3. Idle Kiosk :-

For the idle queue system, it was simulated according to idle waiting line rule where the patient would be send to the kiosk whenever each of the 2 kiosk is available, if both of the kiosks is available, then send the patient to Kiosk#1. The waiting line rule is a little similarly to first come first serve.

In the system, users are prompted to enter the number of patients and choose choice 2 as it is for idle kiosk waiting line rule as shown in Figure 1.2. Then the system will display the Inter-Arrival Time Table and Service Time Table for both Kiosk 1 and Kiosk 2 as shown in Figure 3.1 and 3.2. The inter-arrival time and service time are fixed values. A for loop is created to calculate the inter-arrival time for each patient and it will loop based on the number of patients that the user enters. An if-else statement is created inside the for loop to assign the inter-arrival time of each patient based on the random time generated as shown in Chart 3.1, we have decided to use table array to store all the relevant data.

We have created a while loop and it will stop when all the patients are being served, inside the loop contains a for loop and it will loop based on the number of patients and add the patients into the queue if the arrival time matches the current time. Not only that, there are also other statements such as if the end time of Kiosk 1 and 2 matches the current time, the number of patients in the system will decrement and it will remove the finished patient and assign the kiosk back to the idle state. In the loop there is also another for loop to loop through the patient queue and assign it to the kiosk, if Kiosk 1 is in idle state then assign the patient to Kiosk 1 else to Kiosk 2. Inside the for loop there is also another if statement to assign the correct service time for each kiosk, the end time for both kiosks is calculated by adding current time and service time. If the size of the queue is equal to zero which means there are no more patients, both Kiosks are idle and the current time is larger than the largest arrival time, then the while loop will stop.

For the Simulation Table Part 1, number of patients, random number for inter-arrival time, arrival time, and number of patients in the system were displayed as shown in Figure 3.4. It is stored in the table array and displayed using the for loop.

Another for loop are created to calculate the waiting time and time spend, if kiosk 2 service time is greater than 0 which also means that there is patient inside the kiosk, then we will take

the service begin time of kiosk 2 else kiosk 1 to minus with the arrival time to get the waiting time and total up with total waiting time. Besides, another if statement is if kiosk 1 service time greater than 0 which means there is patient inside the kiosk, then we will take the service time of kiosk 1 else kiosk 2 to calculate the time spend in the system which is adding waiting time with service time, not only that the if else statement is also use to get average service time for both kiosk. There are also other statistics such as total waiting time and total time spend that is calculated here to be displayed at the results evaluation.

For the Simulation Table Part 2, random number of service time, service time, time service begin, time service end for both kiosks, waiting time and time spend in the system, all of the data are stored inside the array and displayed using a for loop.

Lastly is the result evaluation, average waiting time is calculated by taking total waiting time divided by number of patients. Average time spent in the registration is calculated by taking total time spent from both kiosks and divided with the number of patients. Probability that a patient has to wait is calculated by total waiting time divided with the number of patients. Average service time of both kiosks is calculated by adding their service time of each kiosk and divided by how many patients that the kiosk serves.

Flow-Charts :

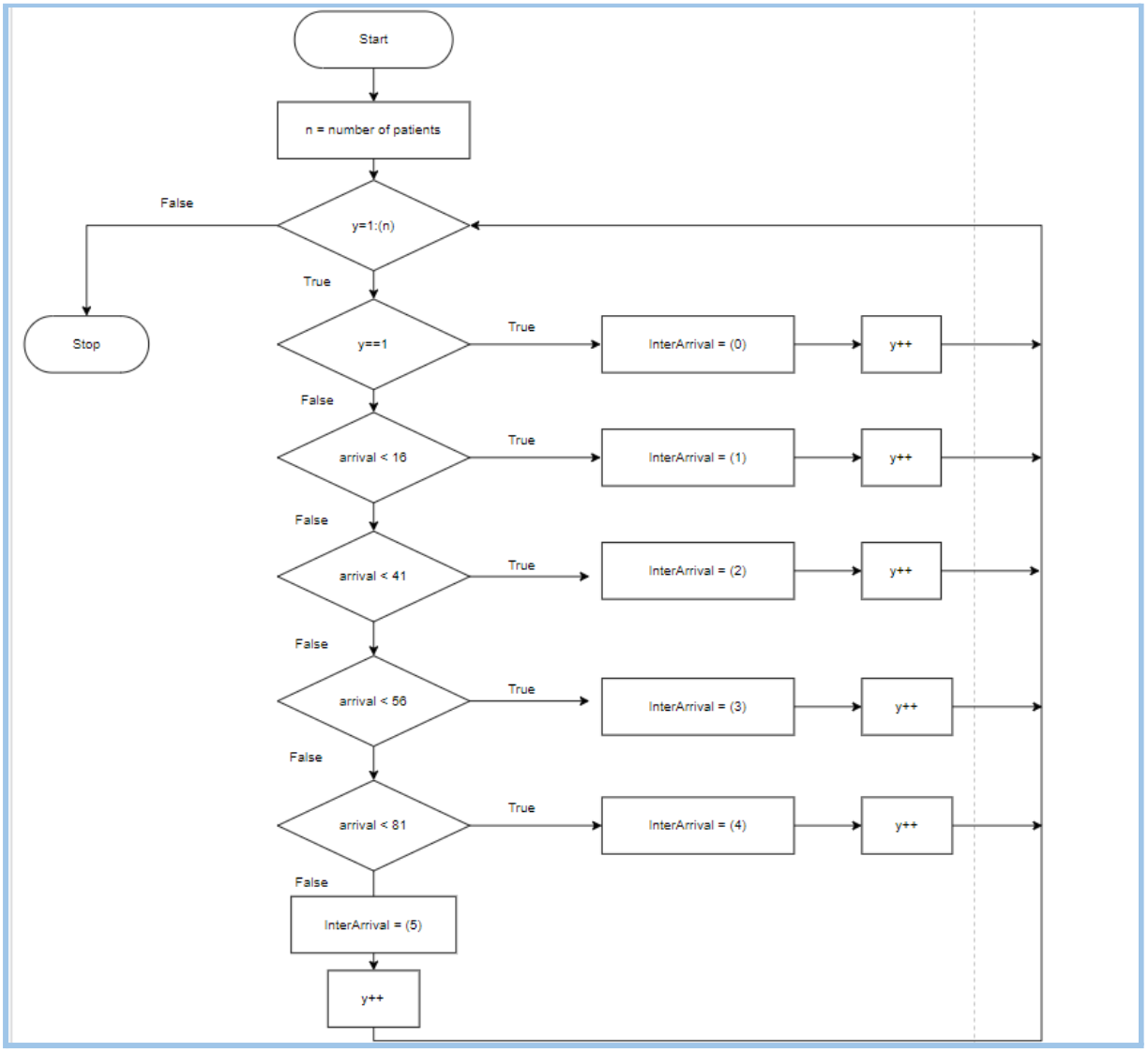

Chart 3.1 Inter-Arrival Time

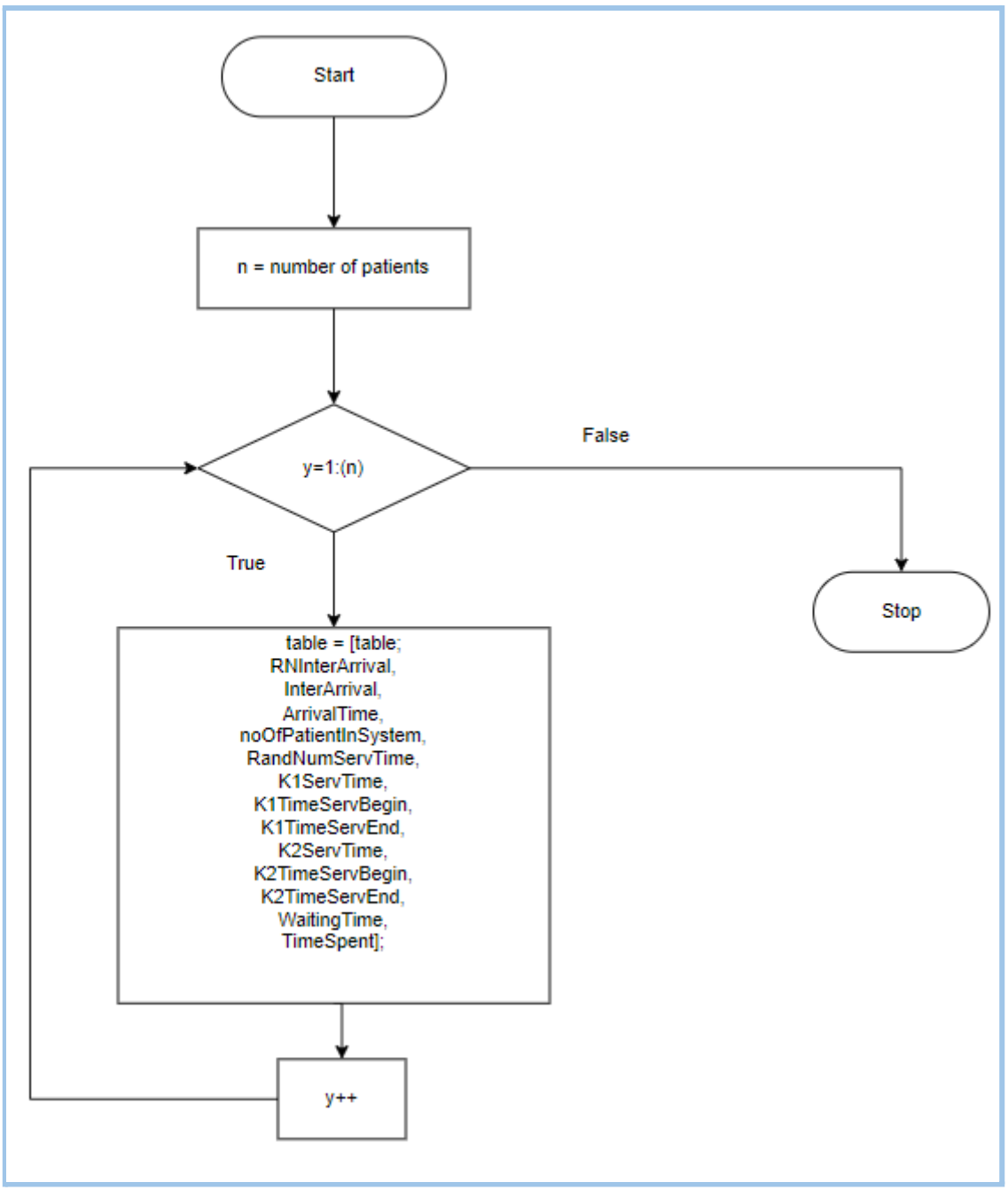

Figure 3.2 Table Array

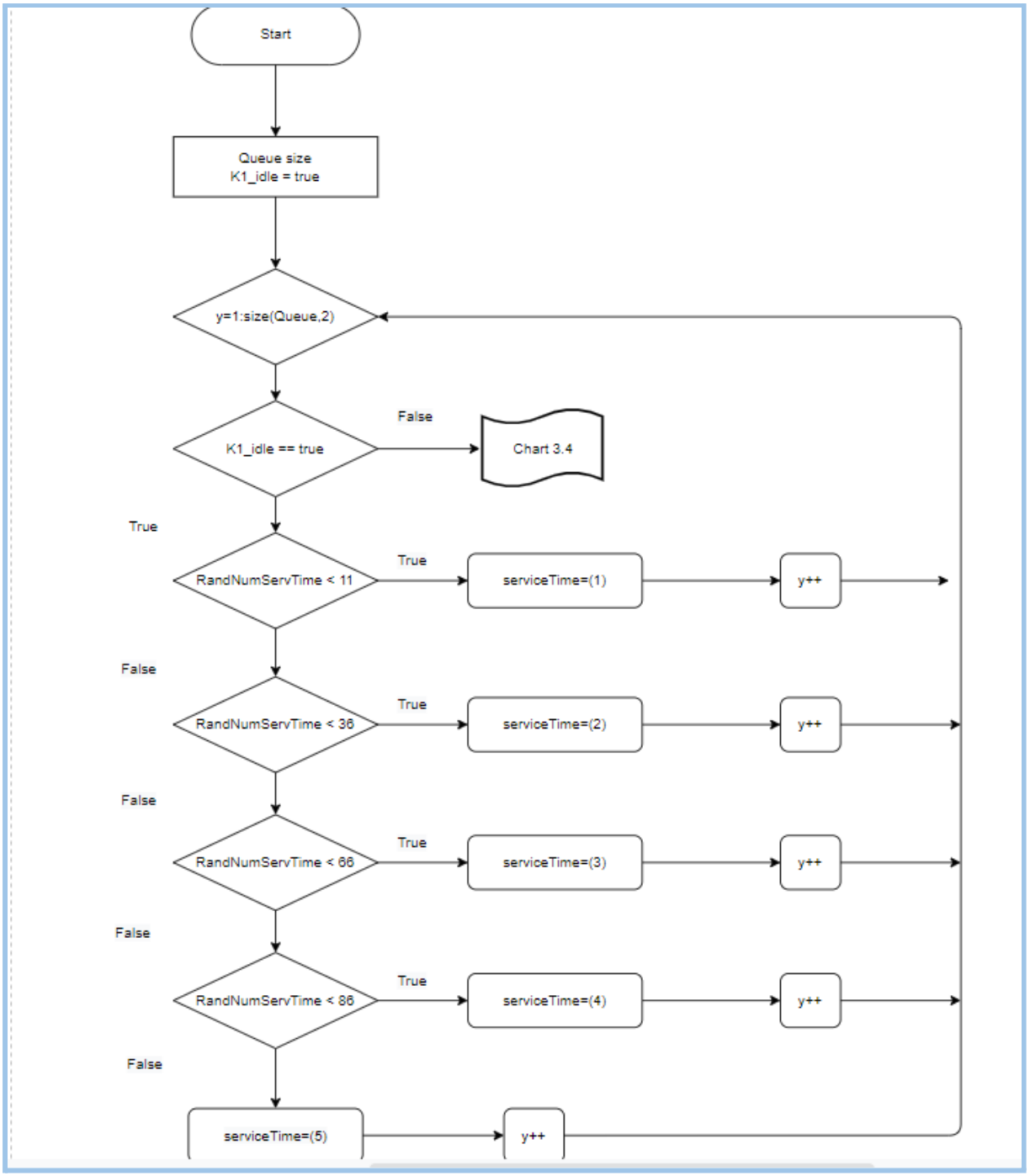

Figure 3.3 Kiosk 1 Service Time Chart

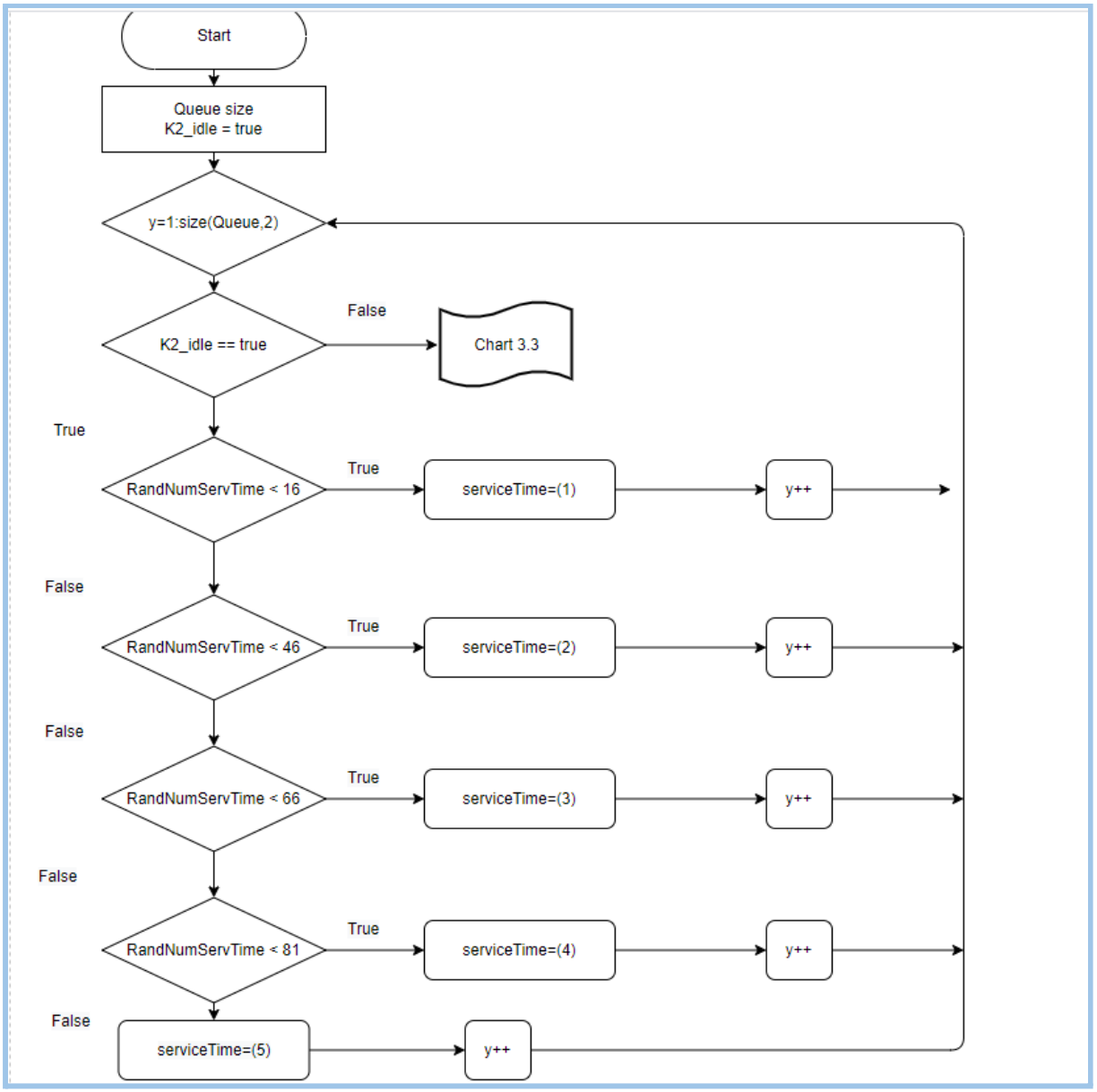

Figure 3.4 Kiosk 2 Service Time Chart

Formulas Used in the System :

The calculation and formulas used in the system as below :

```
%add patient into the queue
 for (y=1:size(table,1))
 AT = table(y,3);

     if( AT == currentTime)
     fprintf('\nPatient %1d ',y);
    fprintf('arrives at minute %1d\n',   AT );

    Queue = [Queue,y];
     table(y,4) = noOfPatientInSystem;
     noOfPatientInSystem = noOfPatientInSystem + 1;
     end



 end

 %if Kiosk 1&2 end time equals to current time No. Patient in the system will decrement
 %remove the finish patient and assign the kiosk back to idle state
 if(K1_end == currentTime)
     K1_idle = true;
     if(K1_end ~= 0)
         noOfPatientInSystem = noOfPatientInSystem -1;
         printf('Depature of Patient at minute %0.0f.\n',  K1_end);
     end
 end
 if(K2_end == currentTime)
     K2_idle = true;
     if(K2_end ~= 0)
         noOfPatientInSystem= noOfPatientInSystem -1;
         printf('Depature of Patient at minute %0.0f.\n',  K2_end);
     end
 end
```

Code 3.1 Number of Patient in the System Calculation

First for loop statement is used to add patients into the queue, the for loop will loop based on the size of the table array, if the arrival time (AT) of patients matches the current time the number of patients in the system (noOFPatientInSystem) will increment. The kiosk end time (K1_end, K2_end) would be calculated by adding the current time when the patient gets service by the kiosk with the service time. If both kiosk end time matches the current time of the system the number of patients in the system will decrement.

```
arrival =floor(rand()*100) + 1;

if(y == 1)
    RNInterArrival=0;
    InterArrival = 0;
    ArrivalTime = 0;
    arrival=0;
end
%display random number for InterArrival time
fprintf('%0.0f ',arrival);
RandNumServTime=floor(rand()*100) + 1;

RNInterArrival = arrival;



if(y == 1)
    InterArrival=(0);

    elseif(arrival < 16)
    InterArrival=(1);

   elseif(arrival < 41)
    InterArrival=(2);

    elseif(arrival < 56)
    InterArrival=(3);

   elseif(arrival < 81)
    InterArrival=(4);

    else
   InterArrival=(5);

end
ArrivalTime = ArrivalTime + InterArrival;
```

Code 3.2 Arrival Time Calculation

The arrival time (ArrivalTime) will be assigned to 0 first as the first patient would not be needed to wait, inside the for loop it will loop based on how many patient that user enters and the if statement above will assign the inter-arrival time (InterArrival) for the patient based on the random inter-arrival time generated. It will then calculate all the arrival time by adding with inter-arrival time of all the patients and assign it to the table array.

```
for y=1:(n)

    RandNumServTime = table(y,5);
    K1ServTime = table(y,6);
    K1TimeServBegin = table(y,7);
    K1TimeServEnd = table(y,8);
    K2ServTime = table(y,9);
    K2TimeServBegin = table(y,10);
    K2TimeServEnd =table(y,11);
    WaitingTime = table(y,12);
    TimeSpend = table(y,13);

    fprintf('%5.0f',y);
    if(RandNumServTime==0)
        fprintf('          -    ');
    else
    fprintf('%13.0f',RandNumServTime);
    end
    if(K1ServTime==0)
        fprintf('             - ');
        fprintf('           - ');
        fprintf('           -');
    else
    fprintf('%13.0f',K1ServTime);
    fprintf('%13.0f',K1TimeServBegin);
    fprintf('%13.0f',K1TimeServEnd);
    end
```

```
table(PatientNo,6) = serviceTime;
table(PatientNo,7) = currentTime;
table(PatientNo,8) = currentTime + serviceTime;
K1_end = currentTime + serviceTime;

printf('Patient %0.0f gets service by K1 at minute
```

Code 3.3 Time Service Begin Calculation

We have assigned the current time when patient enters the kiosk into the table array (table(y,7), (table(y,10)), when the patient get service by the kiosk the current time will be stored inside Kiosk time service begins, so we are able to retrieve the value from the array and display it on the table

```
for(y=1:size(table,1))


    AT = table(y,3);
    K1ServBegin = table(y,7);
    K2ServBegin = table(y,10);

    K1ServT = table(y,6);
    K2ServT = table(y,9);

    if(K2ServT > 0)
        ServBegin = K2ServBegin;
    else
        ServBegin = K1ServBegin;
    end

    WT = ServBegin - AT;
    table(y,12) = WT;
    totalWaitingTime = totalWaitingTime + WT;
```

Code 3.4 Waiting Time In the System Calculations

To calculate the waiting time in the system, first we use a for loop based on the size of the table array and retrieve the data of arrival time (AT) from the table array (table(y,3)) and the service begin time from the 2 kiosks. We define another attribute called ServBegin to get the time from both kiosks. If the service time of kiosk 2 is greater than 0, which means that there is a patient being served by the kiosk else kiosk 1 then we will assign the begin time to ServBegin. To calculate the waiting time of each patient, the service begin time (ServBegin) has to be subtracted with arrival time (AT) and we store it into the table array.

```
if(K1ServT > 0)
     ServiceT = K1ServT;
     AvgServT1 = AvgServT1 + K1ServT;
     ServT1count = ServT1count + 1;
else
     ServiceT = K2ServT;
     AvgServT2 = AvgServT2 + K2ServT;
     ServT2count = ServT2count + 1;

end

TimeSpend = WT + ServiceT;
TotTimeSpend = TotTimeSpend + TimeSpend;
table(y,13) = TimeSpend;
```

Code 3.5 Time Spend In the System Calculations

To calculate the time spent, first we create another attribute (ServiceT) to get the service time from both kiosks. If service time of kiosk 1 is greater than 0 which means there is a patient being served by the kiosk else kiosk 2, then we assign the service time of the running kiosk into ServiceT. To find the time spent we will need to add the waiting time above (WT) with the service time (ServiceT) and stored it into the table array to display at table simulation part 2.

Screenshots :

Screenshot below shows our full program

Inter-Arrival Time Table

Inter-Arrival Time	Probability	CDF	Range
1	0.15	0.15	1-15
2	0.40	0.40	16-40
3	0.55	0.55	41-55
4	0.00	0.80	56-80
5	1.00	1.00	81-100

Figure 3.1 Inter-Arrival Time Table

Service Time for kiosk 1 Table

Service Time for kiosk 1	Probability	CDF	Range
1	0.10	0.10	1-10
2	0.25	0.35	11-35
3	0.15	0.65	36-65
4	0.25	0.85	66-85
5	0.20	1.00	86-100

Service Time for kiosk 2 Table

Service Time for kiosk 2	Probability	CDF	Range
1	0.15	0.15	1-15
2	0.30	0.45	16-45
3	0.20	0.65	46-65
4	0.15	0.80	66-80
5	0.20	1.00	81-100

Figure 3.2 Service Time Table for Kiosk 1&2

```
Arrival & Departure Time Details

Patient 1 arrives at minute 0
Patient 1 gets service by K1 at minute 0

Patient 2 arrives at minute 2
Patient 2 gets service by K2 at minute 2

Patient 3 arrives at minute 3
Depature of Patient at minute 3.
Patient 3 gets service by K1 at minute 3
Depature of Patient at minute 4.

Patient 4 arrives at minute 5
Patient 4 gets service by K1 at minute 5

Patient 5 arrives at minute 6
Depature of Patient at minute 6.
Patient 5 gets service by K2 at minute 6
Depature of Patient at minute 8.

Patient 6 arrives at minute 9
Patient 6 gets service by K2 at minute 9
Depature of Patient at minute 10.
Depature of Patient at minute 12.

Patient 7 arrives at minute 13
Patient 7 gets service by K1 at minute 13
Depature of Patient at minute 16.

Patient 8 arrives at minute 17
Patient 8 gets service by K1 at minute 17

Patient 9 arrives at minute 20
Patient 9 gets service by K2 at minute 20
Depature of Patient at minute 21.
Depature of Patient at minute 22.

Patient 10 arrives at minute 25
Patient 10 gets service by K1 at minute 25
Depature of Patient at minute 28.
```

Figure 3.3 Arrival and Departure Details

SIMULATION TABLE Part 1

Patient	RN for Inter-arrival Time	Inter-arrival Time	Arrival Time	No. of patient in the system
1	0	0	0	0
2	35	2	2	1
3	15	1	3	2
4	18	2	5	1
5	5	1	6	2
6	42	3	9	1
7	76	4	13	0
8	69	4	17	0
9	42	3	20	1
10	85	5	25	0

Figure 3.4 Simulation Table Part 1

SIMULATION TABLE Part 2

		kiosk 1			kiosk 2				
Patient	RN For Service Time	Service Time	Time Service begins	Time Service ends	Service Time	Time Service begins	Time Service ends	Waiting time	Time spend
1	61	3	0	3	-	-	-	0	3
2	76	-	-	-	4	2	6	0	4
3	3	1	3	4	-	-	-	0	1
4	98	5	5	10	-	-	-	0	5
5	25	-	-	-	2	6	8	0	2
6	56	-	-	-	3	9	12	0	3
7	63	3	13	16	-	-	-	0	3
8	75	4	17	21	-	-	-	0	4
9	39	-	-	-	2	20	22	0	2
10	45	3	25	28	-	-	-	0	3

Figure 3.5 Simulation Table Part 2

Results Evaluation:

Average Waiting Time : 0

Average Time Spent in the registration system : 3

Probability that a patient has to wait : 0

Average Service Time of Kiosk 1 : 4.7500

Average Service Time of Kiosk 2 : 2.7500

Figure 3.6 Performance Measures

4. Performance Measures :-

Output of both systems :

```
Results Evaluation:

Average Waiting Time :     0.2000

Average Time Spent in the registration system :     1.7000

Probability that a patient has to wait :     0.2000

Average Service Time of Kiosk 1 : 2

Average Service Time of Kiosk 2 :    1.4000
```

Output 4.1 for Round Robin System for ten patients

```
Results Evaluation:

Average Waiting Time :  0

Average Time Spent in the registration system :     2.4000

Probabilitythat a patient has to wait :  0

Average Service Time of Kiosk 1 :    9.5000

Average Service Time of Kiosk 2 :    2.5000
```

Output 4.2 for Idle Kiosk System for ten patients

Testing both systems for ten patients, as you can see in (*Output 4.1*) the average waiting time and probability is 0.2 while in (*Output 4.2*) the average waiting time and probability is 0, comparing both values from both systems we can see that the waiting time and probability for (*Output 4.2*) is smaller. However, the average time spent in the registration system for (*Output 4.1*) is 1.7 and (*Output 4.2*) 2.4, the average time spent in the registration system for (*Output 4.2*) is larger by 0.7 compared to (*Output 4.1*) . The average service time for kiosk 1 in (*Output 4.1*) is 2 and in (*Output 4.2*) is 9.5, as you can see there is a big difference where (*Output 4.2*) is 7.5 . However, the average service time for kiosk 2 in (*Output 4.1*) is 1.4 and

in (*Output 4.2*) is 2.5, as you can see that average service time for kiosk 2 in (*Output 4.2*) is larger by 1.1 compared to the average service time for kiosk 2 in (*Output 4.1*).

```
Results Evaluation:

Average Waiting Time :     0.6538

Average Time Spent in the registration system :     2.4615

Probability that a patient has to wait :     0.6538

Average Service Time of Kiosk 1 :    2.5385

Average Service Time of Kiosk 2 :    2.3846
```

Output 4.3 for Round Robin System for twenty-six patients

```
Results Evaluation:

Average Waiting Time :  0

Average Time Spent in the registration system :     3.1538

Probabilitythat a patient has to wait :  0

Average Service Time of Kiosk 1 :    5.5556

Average Service Time of Kiosk 2 :    3.5556
```

Output 4.4 for Idle Kiosk System for twenty-six patients

Both systems round robin and idle kiosk were tested with the same amount of patients which is 26 patients. As shown from the image above output 4.3 and output 4.4 we can see that they have different results even though with the same input. Average time of the round robin system is 0.6538 where average waiting time of idle kiosk is at 0. This means the idle kiosk system is more efficient since the waiting time is zero. The average time spent in the registration system for both systems is near where round robin is 2.4615 shorter than idle kiosk which is 3.1538. Since the average waiting time for an idle kiosk is 0, then the probability of waiting is zero while the probability

of waiting for round robin is as the same as the average waiting time which is 0.6538. The average service time of kiosk 1 is 2.5385 and service time of kiosk 2 is 2.5385 for round robin system is less than idle kiosk average service time for kiosk 1 and 2 which is 5.5556 and 3.5556.