TMA1301 COMPUTATIONAL METHODS

TT02

Assignment Report

Concert Ticket System Simulator

LECTURER : MS LIM SIEW LING

TUTOR : DR TONG HAU LEE

PREPARED BY

CASEY TEH QI SHI 1171103211 1171103211@student.mmu.edu.my

CHUA BING QUAN 1181303556 1181303556@student.mmu.edu.my

LIEW KUAN YUNG 1191301064 1191301064@student.mmu.edu.my

KOH SHI JING 1171103504 1171103504@student.mmu.edu.my

Table of Contents

1.0 Introduction

The team chose to create a two day concert ticket purchasing simulation system that sells 2 types of tickets, which are normal tickets and VIP tickets. For the purpose of the project, there are four counters in total where one of those is only opened for members while the rest are for non-members. The customers will queue up in a line according to their arrival time and will proceed to any counter(s) that are free to serve the customers. The simulation will end when all the customers have been served.

Users will be asked to input the number of customers for the simulation as well as the type of random numbers generators like Linear Congruential Generators(LCG), Random Variate Generator for Exponential Distribution and Random Variate Generator for Uniform Distribution. The team aims to create a clear and accurate algorithm that can solve the mathematical problem in hand efficiently.

2.0 Simulator Description

1. The user will input the number of customers for the simulation.

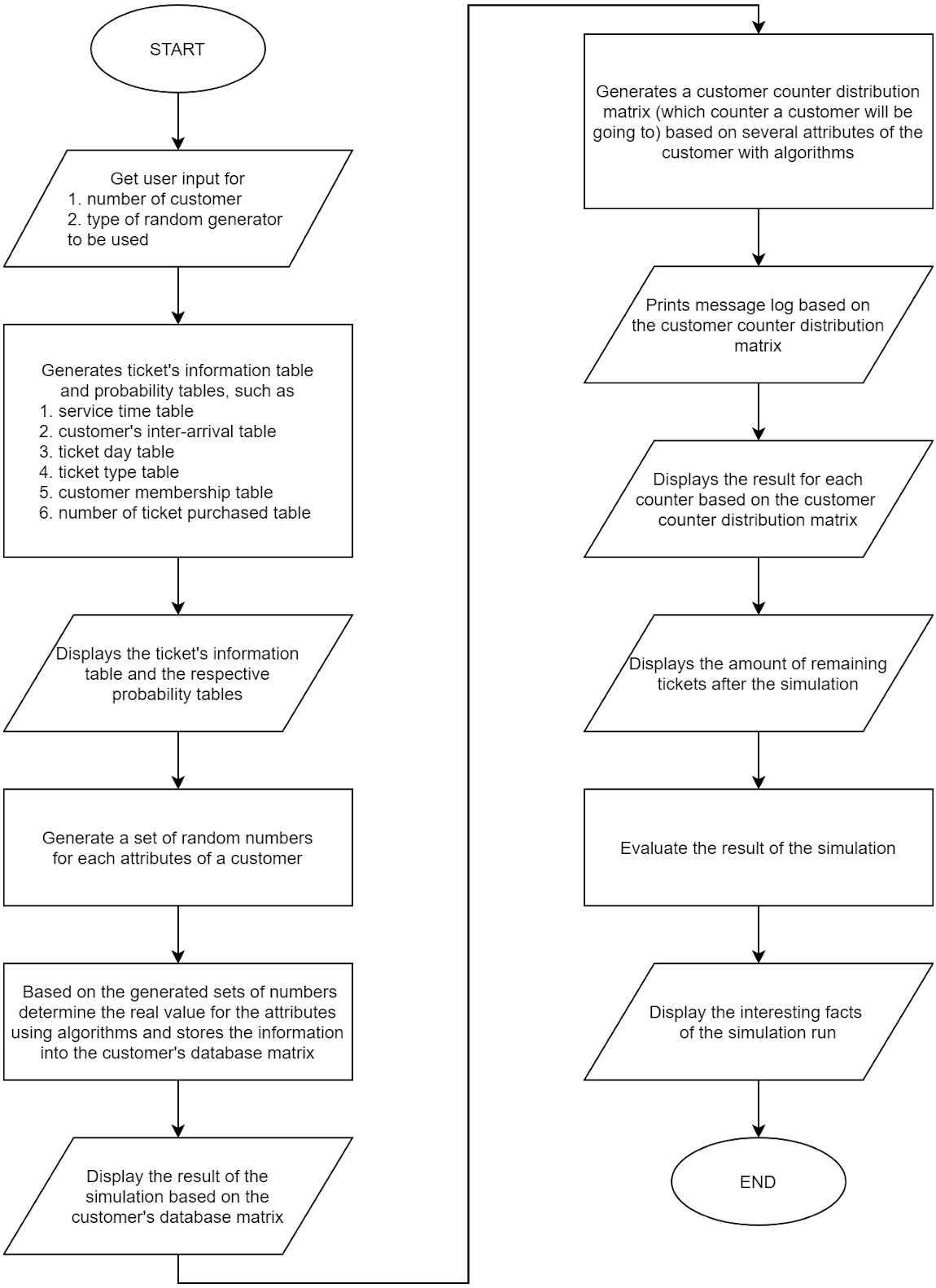
2. The user is able to choose one out of three random number generators, Linear Congruential Generators(LCG), Random Variate Generator for Exponential Distribution and Random Variate Generator for Uniform Distribution, to generate the range for customers interarrival time, ticket day for the concert, ticket type, number of ticket and counter service time.

3. The simulator will display all the tables with the probabilities, such as Ticket Information Table, Service Time Table, Interarrival Time Table, Ticket Slot Table, Ticket Type Table and Membership Table.

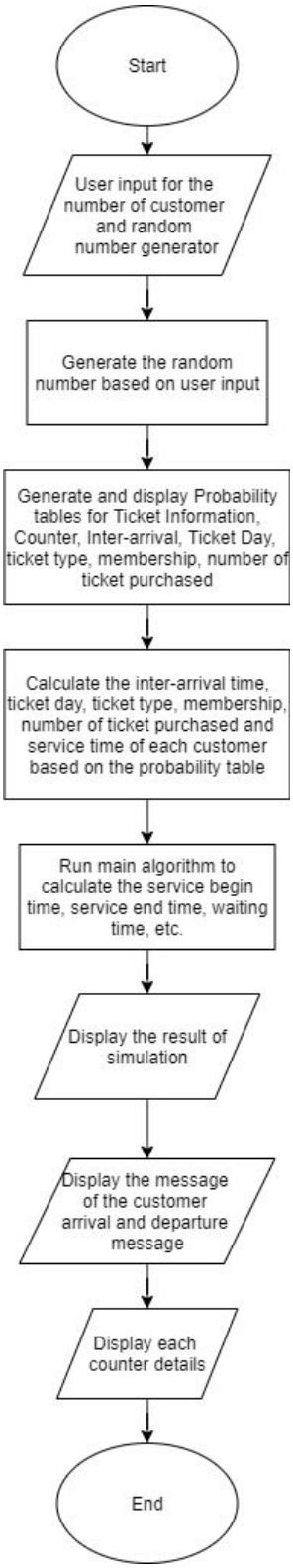
4. The result of simulation is printed followed by the messages of the arrival and departure of every customer with the number of tickets they bought, the ticket types and the time when service started.

5. Each arriving customer status and the status of each counter will be printed. Then, the results in every counter are displayed and the system will show number of tickets left at the end of simulation. Finally, a few evaluations on the simulation run will be printed.

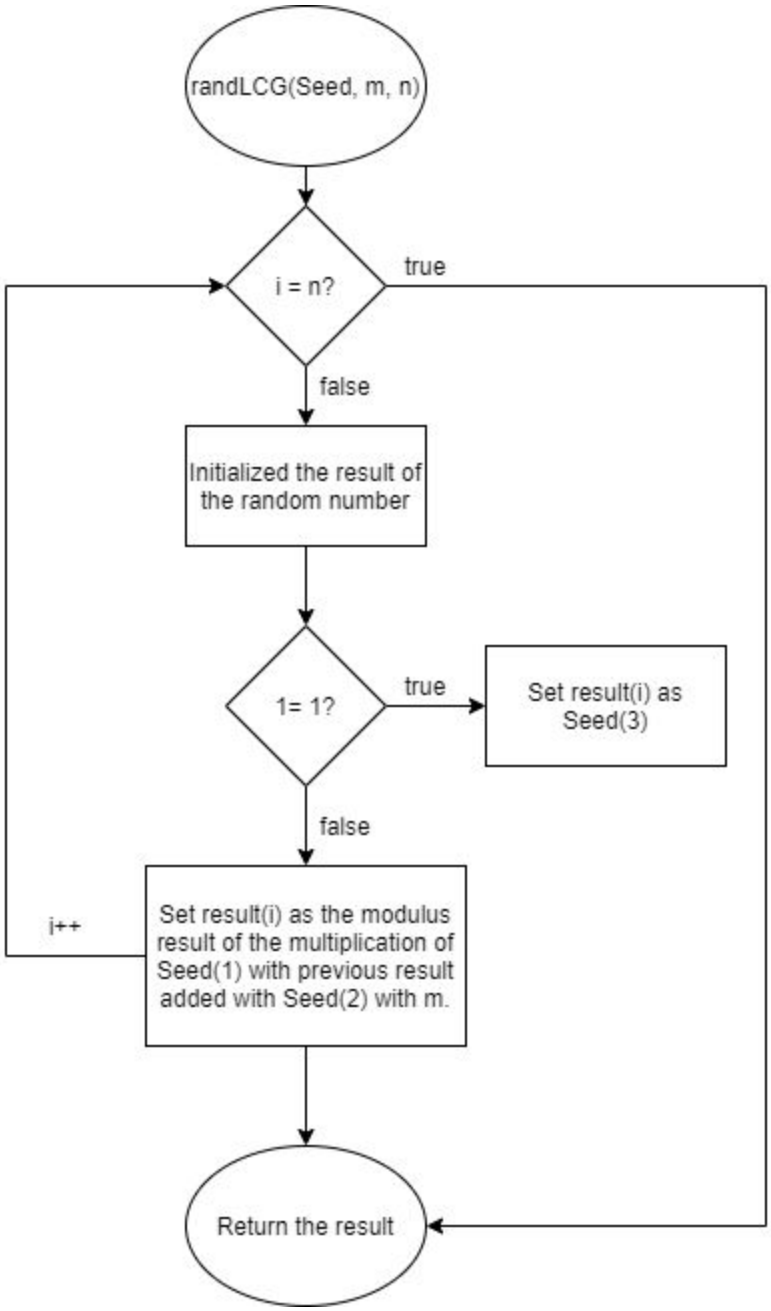
3.0 Program Flowchart



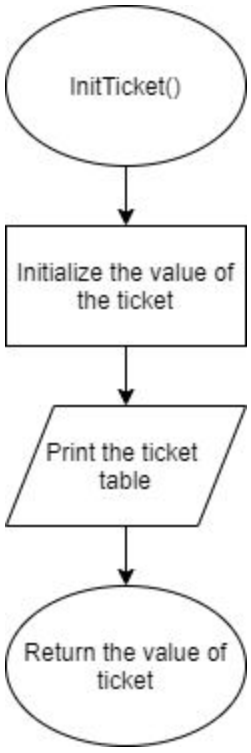
3.1 Main function(Main.m)



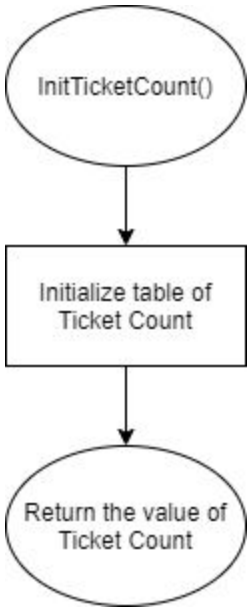
3.2 Generate the random number(randLCG.m)



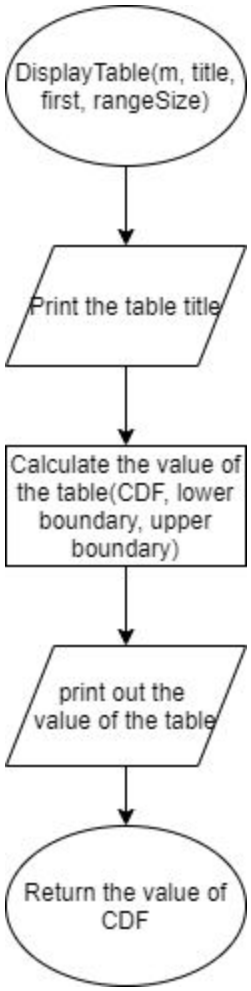
3.3 Print out the table for the tickets(InitTicket.m)



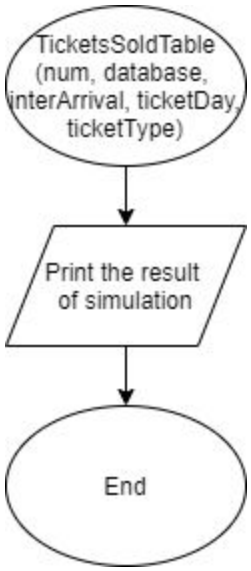
3.4 Initialize the number of ticket(InitTicketCount.m)



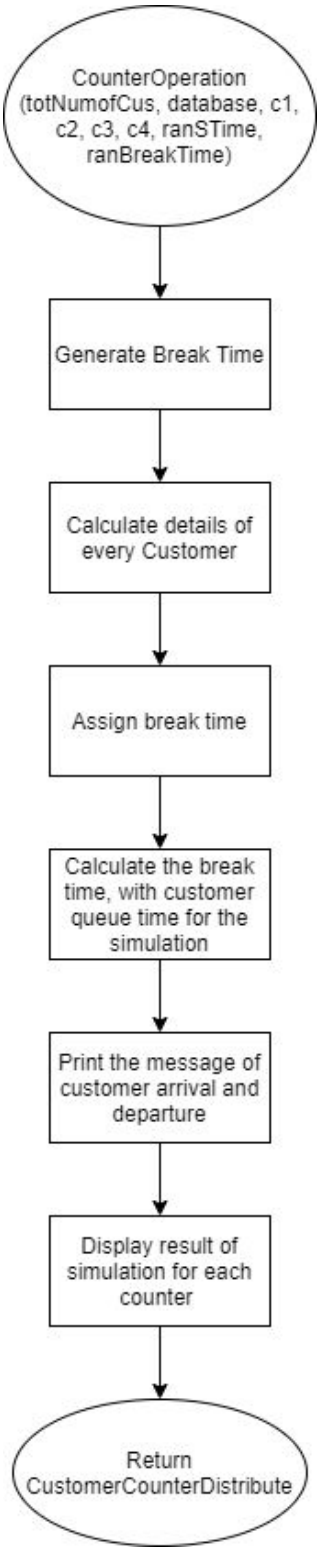
3.5 Print out function for the table (DisplayTable.m)



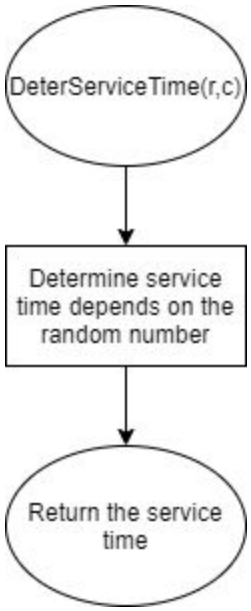
3.6 Display the result of simulation(TicketsSoldTable.m)



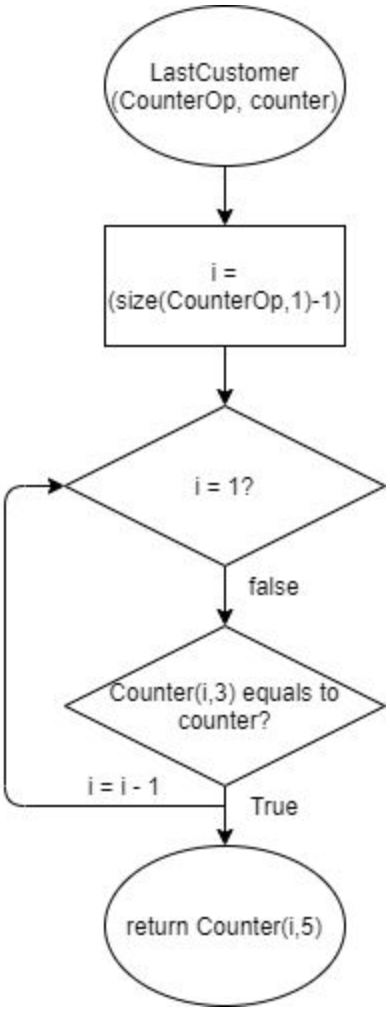
3.7 Counter Operation(CounterOperation.m)



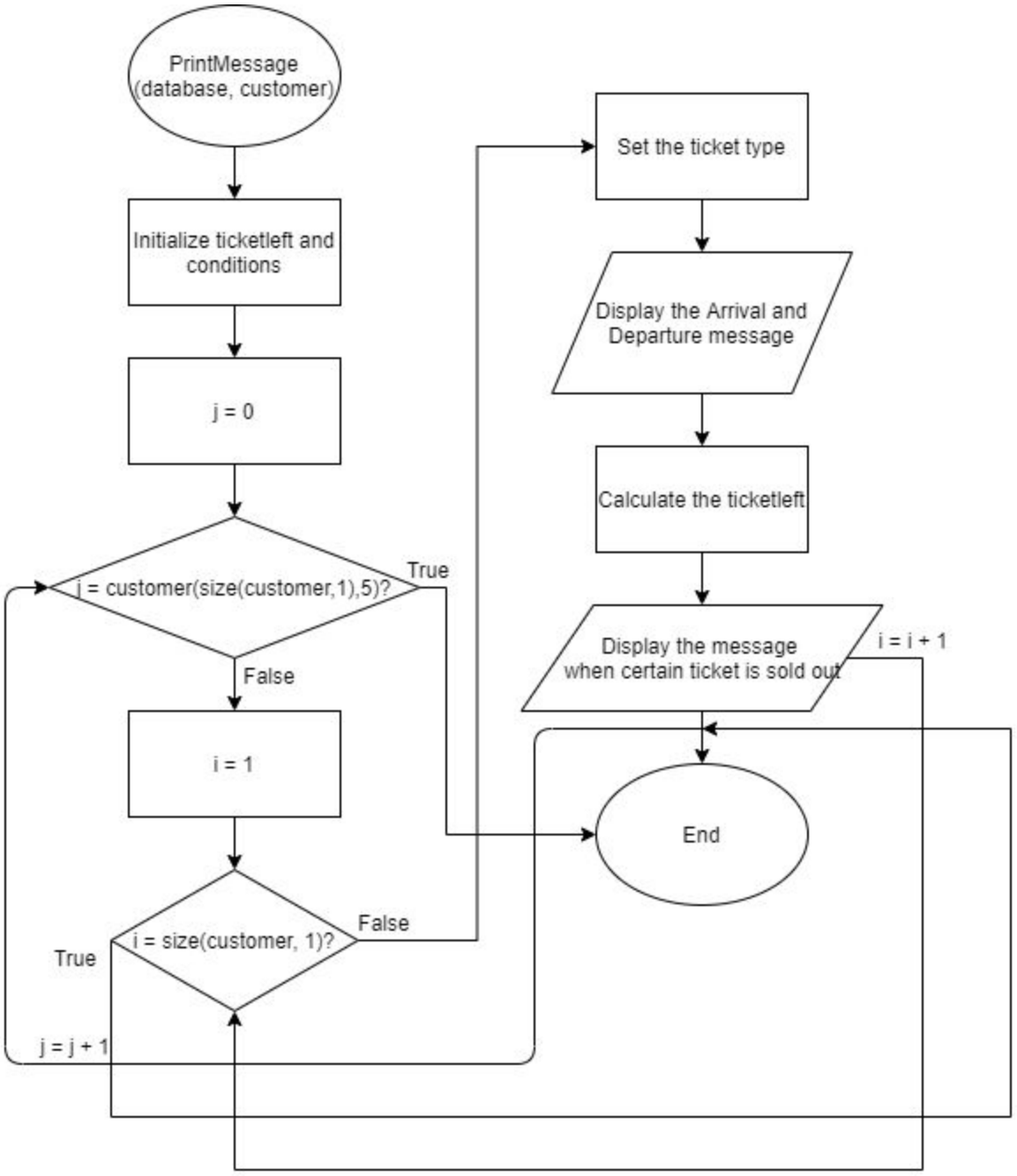
3.8 Determine the service time(DeterServiceTime.m)



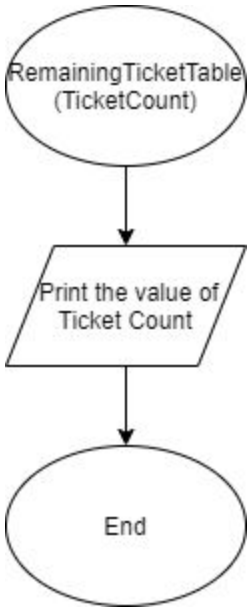
3.9 Calculate the last customer in the counters(LastCustomer.m)



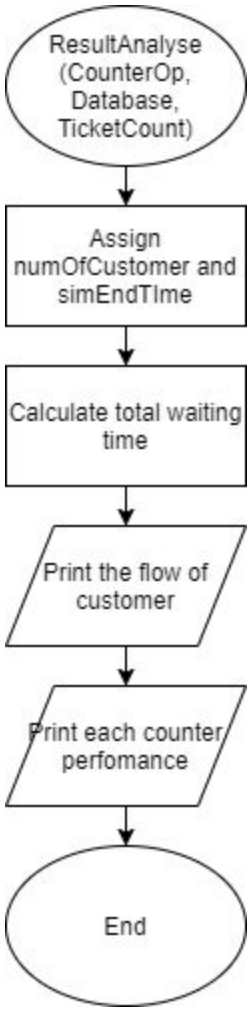
3.10 Display the message on the customer arrival and departure(PrintMessage.m)



3.11 Print out the remaining ticket after simulation(RemainingTicketTable.m)



3.12 Analysis the result after simulation done(ResultAnalyse.m)



Summary on all the functions available in this project:

Main.m - The main function of this program.

randLCG.m - Random number generator for Linear Congruential Generator.

InitTicket.m - Initiate the matrix of ticket and print the table.

InitTicketCount.m - Initiate the matrix of ticket and ticket price, use for tracking the number of tickets later.

DisplayTable.m - Accepting table matrices and their name, then printing them into tables for displaying.

TicketsSoldTable.m - Displays the results of the simulation.

CounterOperation.m - Accepts the Database matrix and determines which counter should a customer go and generates the CustomerCounterDistribute which holds all the needed information for the simulation.

DeterServiceTime.m - Determine the service time for each customer depends on their random number generated corresponding with the respective counter.

LastCustomer.m - Determine the end service time of the last customer that has been served in that counter.

PrintMessage.m - Print out the message upon customer arrival and departure.

RemainingTicketTable.m - Show the remaining ticket after simulation ends.

ResultAnalyse.m - Analyse the result of simulation such as Flow of Customer and Individual Counter Performance and Overall Performance.

4.0 Main Codes and Algorithms

4.1 Random Number Generators

The team has chosen three types of random number generators for this project, which are Linear Congruential Generator, Random Variate Generator for Exponential Distribution and Random Variate Generator for Uniform Distribution.

1) Linear Congruential Generators (LCG)

```
% file: randLCG.m
function Output = randLCG(Seed, m, n)
    result = [];
    for i = 1:n
        if i == 1
            result(i) = Seed(3);
        else
           result(i) = mod((Seed(1) * result(i - 1) + Seed(2)), m);
        end
    end
    Output = result;
```

```
% file: Main.m
RanServiceTime                                                    =
mod(randLCG(ceil(rand(1,3)*100),143,numOfPeople),99)+1;
...
```

The above codes are used for generating n numbers of random numbers referencing the base Linear Congruential Generator formula shown below:

$$X_n = (aX_{n-1} + c) \pmod{m}$$

2) Random Variate Generator for Exponential Distribution

For this random number generator, the FreeMat built-in Random Variate Generator for Exponential Distribution where it is used as

```
randexp(lambda);
%lambda is a vector
```

```
% file: Main.m
RanServiceTime = mod(round(randexp(100*ones(1,numOfPeople))),99)+1;
...
```

The above codes are used for generating n numbers of random numbers referencing the base Random Variate Generator for Exponential Distribution formula shown below:

$$X_i = (-1/\lambda)\ln(1 - R_i)$$

3) Random Variate Generator for Uniform Distribution

The built-in function was also used for this random number generator where it is used as

```
y = randi(low,high);

% low and high is an array of integers
```

```
% file: Main.m
RanServiceTime=randi(zeros(1,numOfPeople),
                100*ones(1,numOfPeople));
...
```

These three random number generators will generate a series of random numbers which are passed as a list of random number to seven different variables as below:

```
RanServiceTime = ...
RanInterArrival = ...
RanTicketDay = ...
RanTicketType = ...
RanMembership = ...
RanNumberOfTicketPurchased =  ...
RanCasherBreak = ...
```

The generated list of random numbers will later be used as attributes for each customer.

4.2 Implementation of Probability Tables

Matrices were used to initialize all the probability tables such as all the probability of all counters' service time, tickets day and type, membership and inter arrival time and casherBreakTime.

Example of this is where we initialize the probability of service time for Counter 1.

```
Counter1 =
[3, 0.2 ;
 4, 0.3 ;
 5, 0.3 ;
 6, 0.2 ];
```

The above code will be passed into our table displaying function (DisplayTable.m) and be added a third column, which holds the Cumulative Distribution of the probability for later use.

4.3 Further Implementation for other Features of Simulation

To make effective access to data, after generating n numbers (user input of number of customers), a bigger matrix is generated named Database which stores the Inter Arrival numbers, Arrival, Day, Type, Membership, Number, Paid. This matrix mainly holds all the information for each customer in one simulation run.

The assignment of the values in this Database matrix are implemented using these codes, for instance, this is the assignment of the Ticket Day:

```
% file: Main.m
%    1     2     3      4      5      6      7
% [ IAT | Arr | Day | Type | Member | No | Paid ]
for i = 1:numOfPeople
        if(i == 1) % for first row
            Database(i,1) = 0;
            Database(i,2) = 0;
```

```
        else
            ...
        end
        ...
        for d = 1:2
            if(RanTicketDay(i) <= (TicketDay(d,3))*100)
                Database(i,3) = TicketDay(d,1);
                break
            end
        end
        ...
...
```

Ticket Day Table			
Ticket Day	Probability	CDF	Range
1	0.550000	0.550000	0 - 55
2	0.450000	1.000000	56 - 100

For instance, if RanTicketDay(1) = 78

Checking row by row,

Row 1: RanTicketDay(1) < (TicketDay(1,3))*1000 -> False

Row 2: RanTicketDay(1) < (TicketDay(2,3))*1000 -> True

Therefore Database(1,3) = TicketDay(2,1) = 2

4.4 Determining the Service Time

```
% File: DeterServiceTime.m

function Output = DeterServiceTime(r,c)
    % r: generated random number
    % c: counter matrix

    row = size(c,1); % number of rows for the matrix

    for i = 1:row
        if( r <= (c(i,3))*100)
            Output = c(i,1);
            break
        end
    end
```

We use the codes above to determine the service time by passing in the generated random number and matrix of the counter. We get the number of rows of the matrix by the code snippet below.

```
row = size(c,1);
```

Then we loop each row to find the probability of the service time multiplied by 100. If the random number generated is smaller than the row's probability multiplied by 100, then that is the service time determined for the specified random number.

4.5 Distributing customers to which counters

This function gets the number, the database matrix, counter1 matrix, counter2 matrix, counter3 matrix and counter4 matrix. This function generates another big matrix where the attributes of the columns are shown below:

```
% File: CounterOperation.m
   1              2               3              4              5
[ index | ori arrival time | counter no. | service time | services
                     6                  7
start time | services end time | total time in queue |
      8
RanServiceTime]

CustomerCounterDistribute = [];
```

This simulation has 4 counters, where the first three are for non-member customers and the last is exclusively for customers who are members to queue. The priority of the waiting line for non-members is Counter 1 - Counter 2 - Counter 3. For example, if Counter 1 is currently busy with a customer and Counter 2 is free, the customer will go to Counter 2 or so on. If all of them are serving the previous customer while a new customer arrives, he will be queuing at the counter with the serving time coming to an end. For members, they will only be queuing at Counter 4.

Before the main loop begins, our team assign the generated number for the counter break with their respective values and obtain the total number of breaks that is going to happen in the simulation. Counter break is where the employee of the counter went for a short break, so the counter will be temporarily closed and customers won't be queuing there for the time being.
The number of break occurrences will be added to the number of customers, resulting in the total number the main loop will be looping.

The first customer of the simulation is a special case. Unlike other customers, the first customer does not have an inter arrival time, which means he will be arriving at time 0. There is no previous customer so he will be going to Counter 1 or Counter 4 depending on whether he is a member or not. Other values of this customer are assigned automatically.

```
%%% first customer special case
CustomerCounterDistribute(1,1) = 1;
CustomerCounterDistribute(1,2) = database(1,2);
if database(1,5) == 1; %%% is member
      CustomerCounterDistribute(1,3) = 4;
      CustomerCounterDistribute(1,4) =
      DeterServiceTime(ranSTime(1),c4);
else

      CustomerCounterDistribute(1,3) = 1;
      CustomerCounterDistribute(1,4) =
      DeterServiceTime(ranSTime(1),c1);
end
CustomerCounterDistribute(1,5) = 0;
CustomerCounterDistribute(1,6) = CustomerCounterDistribute(1,4);
CustomerCounterDistribute(1,7) = 0;
CustomerCounterDistribute(1,8) = ranSTime(1);
```

The main loop will then start from 2 until the end (number of customers plus the number of breaks that will be happening). At the beginning of the loop, we check whether a counter is going on a break. If that happens, we will assign the index of the matrix row as 0 and arrival time as -1, indicating that a break happens at that time.

```
 % if break happens
for j = 1:numOfBreak
      if i == breakDatabase(j,2)
            CustomerCounterDistribute(i,1) = 0;
            CustomerCounterDistribute(i,2) = -1;
            CustomerCounterDistribute(i,3) =
            CustomerCounterDistribute(i-1,3); %%Place at last
            counter
```

```
            CustomerCounterDistribute(i,4) = breakDatabase(j,1);
            CustomerCounterDistribute(i,5) =
            CustomerCounterDistribute(i-1,6);
            CustomerCounterDistribute(i,6) =
            CustomerCounterDistribute(i,4) +
            CustomerCounterDistribute(i,5);
            CustomerCounterDistribute(i,7) = 0;
            CustomerCounterDistribute(i,8) = 0;
            casherBreakChecking = casherBreakChecking +1 ;
            breakDatabase(j,2) = totalNumOfCus + 100 + j;
            gotBreak = 1;
            break;
      end
      gotBreak = 0;
end

if gotBreak == 1
      continue;
end
```

The loop ends after we handle the break.

If no break happens, the customer will be able to go to the counter to queue normally. We check if the arriving customer is a member or not, if not, he will be assigned to Counter 4 for queuing. For members, they will queue according to the algorithm that was stated on top.

Customers will be automatically assigned to a Counter if the counter is free, following the order of Counter 1 - Counter 2 - Counter 3.

First, we check if Counter 1 is occupied. If yes, we will then check Counter 2. If Counter 2 is also busy, then we will check for Counter 3. If all of the counters are busy, we will compare the waiting time of each counter and assign the customer to the one with the shortest waiting time.

The loop will end after all of the customers have been served, hence the CustomerCounterDistribute matrix is completed.

We use the information that we gathered to print the result table of each counter using another loop.

4.6 Getting the service time of the last customer

In this function (LastCustomer.m), we get the service time of the previous customer by passing in the customer counter distribute matrix and the counter number where we loop i from size of the counter minus by one, decreasing by 1 until 1. If the counter number is the counter, we return the service time.

```
% File: LastCustomer.m
function Output = LastCustomer(CounterOp,counter)
    % CounterOp: CustomerCounterDistribute
    % counter: number of counter
    notFound = 0;
    for i = size(CounterOp,1)-1:-1:1     %%% size() - 1 to exclude
                                         itself
        if(CounterOp(i,3) == counter)
           if CounterOp(i,6) > 0
                notFound = 1;
                Output = CounterOp(i,6); %%% returns the service
                                         ending time of same
                                         counter but previous
                                         customer
                break;
           end
        end
    end

    if notFound == 0
      Output = 0;
    end
```

If there was no last customer, as in the customer coming to this customer is the first, the function will return 0.

4.7 Printing of Messages

We use the 2 implemented matrix described above and access their data to create the messages and loop through the number of customers.

```
function PrintMessage(database, customer)
% database :
% [ IAT | Arr | Day | Type | Member | No | Paid ]
% customerCounterDistribute :
   1              2              3              4              5
[ index | ori arrival time | counter no. | service time | services
                     6                  7
start time | services end time | total time in queue |
      8
RanServiceTime]
simEndTime = 0;

% finding the time the simulation ends

for i = size(CounterOp,1)-1:-1:size(CounterOp,1)-6
      if CounterOp(i,6) > simEndTime
            simEndTime = CounterOp(i,6);
      end
end

printf('Counter 1, Counter 2, Counter 3, Counter 4 are in
operation.\n');
printf('Counter 4 is exclusively for members only.\n');
for i = 0:simEndTime
      for j = 1:size(CounterOp,1)
            if i == CounterOp(j,2) % arrival
                  printf('Arrival of Customer %d at minute %d.
                  \n',CounterOp(j,1),CounterOp(j,2));
            end
            if i == CounterOp(j,5) && CounterOp(j,2) > -1 % service
            time starts
            cusNo = CounterOp(j,1);

                  if database(cusNo,4) == 1
```

```
                    tickettype = 'Normal';
                else
                    tickettype = 'VIP';
                end
                printf('Service for Customer %d starts at minute
                %d. \n',cusNo,CounterOp(j,5));
                printf('Customer %d bought %d Day%d %s tickets
                from Counter %d.
                \n',cusNo,database(cusNo,6),database(cusNo,3),tick
                ettype,CounterOp(j,3));
            end
            if i == CounterOp(j,5) && CounterOp(j,2) == -1 %
            counter close
                printf('Counter %d takes a %d minute break at
                minute %d.
                \n',CounterOp(j,3),CounterOp(j,4),CounterOp(j,5));

            end
        end
end
```

4.8 Displaying/Printing of tables

A function DisplayTable was made specifically for printing of all probability and information tables and was made dynamically. Arguments such as the matrix, title, the first column name, and the size of range are passed into this function.

```
% File: DisplayTable.m

function Output = DisplayTable(m, title, first, rangeSize)
fprintf('\n+----------------------------------------+\n')
if(size(title) <25)
    fprintf('| %37s %18s\n' , title , '|' )
else
    fprintf('| %45s %10s\n' , title , '|' )
end

fprintf('+-------------------------------------------------+\n')
fprintf('| %13s %40s\n', first , '| Probability | CDF |Range   |')
printf('+--------------+----------+-----------+------------+\n')

A = size(m,1);
CDF = 0;
upperBoundary = 0;
lowerBoundary = 0;

for i = 1:A
    CDF = CDF + m(i,2);
    lowerBoundary = CDF * rangeSize;
    printf('| %2d | %f | %f |%4d - %4d |\n', m(i,1),m(i,2), CDF,
    upperBoundary, lowerBoundary)
    upperBoundary = lowerBoundary + 1;
    m(i,3) = CDF;
end
printf('+-------------------------------------------------+\n')
Output = m;
```

4.9 Result analyse

This function takes in the generated matrices: Database, CustomerCounterDistribute from Counter Operation and TicketCount and calculate a few interesting facts regarding the simulation run. Users may analyse the simulation with the sets of calculated results.

We have provide calculation for:

1. Customer's average interarrival time = $\frac{Total\ Interarrival\ Time}{Number\ of\ Customers}$
2. Customer's average arrival time = $\frac{Total\ Arrival\ Time}{Number\ of\ Customers}$
3. Average service time for each counter = $\frac{Total\ Service\ Time\ of\ the\ Counter}{Number\ of\ Customers\ going\ to\ the\ Counter}$
4. Sales made by each counter = *Total Amount of Sales Made*
5. Percentage of time each counter is busy = $\frac{Total\ Service\ Time\ of\ the\ Counter}{Total\ Time\ of\ the\ Simulation}$
6. Average waiting time for each customer = $\frac{Total\ Waiting\ Time\ of\ all\ Customers}{Number\ of\ Customers}$
7. Probability that a customer needs to wait = $\frac{Number\ of\ Customers\ that\ needs\ to\ wait}{Number\ of\ Customers}$
8. Average time a customer spends in the system = $\frac{Total\ Time\ Spent\ in\ the\ System\ of\ each\ Customer}{Number\ of\ Customers}$
9. Percentage of sale for each ticket type = $\left(1 - \frac{Number\ of\ Tickets\ Sold}{Number\ of\ Tickets\ Originally}\right) * 100\%$

5.0 Simulation Inputs and Outputs

5.1 Inputs

1) Input number of customers for the concert

```
Please input the number of people (at least 10): 20
```

2) Input type of Random Number Generator

```
1: LCG
2: Exponential Distribution
3: Uniform Distribution
Please enter the type of random number generator to use: 3
```

5.2 Outputs

1) Ticket Information Table

Ticket Information			
Day	Normal	VIP	Total
1	35	15	50
2	35	15	50

2) Counter 1 Service Time Probability Table

Counter 1(Non-Member)			
Service time	Probability	CDF	Range
3	0.200000	0.200000	0 - 20
4	0.300000	0.500000	21 - 50
5	0.300000	0.800000	51 - 80
6	0.200000	1.000000	81 - 100

3) Counter 2 Service Time Probability Table

Counter 2(Non-Member)			
Service time	Probability	CDF	Range
4	0.350000	0.350000	0 - 35
5	0.250000	0.600000	36 - 60
6	0.200000	0.800000	61 - 80
7	0.100000	0.900000	81 - 90
8	0.100000	1.000000	91 - 100

4) Counter 3 Service Time Probability Table

Counter 3(Non-Member)			
Service time	Probability	CDF	Range
2	0.300000	0.350000	0 - 30
3	0.280000	0.580000	31 - 58
4	0.250000	0.830000	59 - 83
5	0.170000	1.000000	84 - 100

5) Counter 4 Service Time Probability Table

Counter 4(Member)			
Service time	Probability	CDF	Range
4	0.200000	0.200000	0 - 20
5	0.200000	0.400000	21 - 40
6	0.250000	0.650000	41 - 65
7	0.200000	0.850000	66 - 85
8	0.150000	1.000000	86 - 100

6) Inter arrival Time Probability Table

Interarrival Time			
Inter-arrival time	Probability	CDF	Range
1	0.125000	0.125000	0 - 125
2	0.125000	0.250000	126 - 250
3	0.125000	0.375000	251 - 375
4	0.125000	0.500000	376 - 500
5	0.125000	0.625000	501 - 625
6	0.125000	0.750000	626 - 750
7	0.125000	0.875000	751 - 875
8	0.125000	1.000000	876 - 1000

7) Ticket Day Probability Table

Ticket Day Table			
Ticket Day	Probability	CDF	Range
1	0.550000	0.550000	0 - 55
2	0.450000	1.000000	56 - 100

8) Ticket Type Probability Table

Ticket Type Table			
Ticket Type	Probability	CDF	Range
1	0.750000	0.750000	0 - 75
2	0.250000	1.000000	76 - 100

9) Membership Probability Table

Membership Table			
Membership	Probability	CDF	Range
1	0.200000	0.200000	0 - 2
2	0.800000	1.000000	3 - 10

10) Number Of Ticket Purchased Probability Table

Number of Ticket Purchased Table			
Number of ticket purchased	Probability	CDF	Range
1	0.150000	0.150000	0 - 15
2	0.600000	0.750000	16 - 75
3	0.250000	1.000000	76 - 100

11) Cashier Break TIme Table

Cashier Break Time			
Time Spent	Probability	CDF	Range
6	0.400000	0.400000	0 - 40
7	0.250000	0.650000	41 - 65
8	0.150000	0.800000	66 - 80
9	0.100000	0.900000	81 - 90
10	0.100000	1.000000	91 - 100

12) Random Numbers Generated

```
The random numbers for inter-arrival time are:
-------------------------------------------------------------------------------------------
376 218 488 577 801 403 259 90 746 542 139 826 969 134 486 224 184 312 700 175
-------------------------------------------------------------------------------------------

The random numbers for service time are:
-------------------------------------------------------------------------------------------
67 86 18 3 88 48 51 75 57 56 63 30 67 61 44 86 96 19 41 62
-------------------------------------------------------------------------------------------

The random numbers for membership time are:
-------------------------------------------------------------------------------------------
1 9 7 5 4 8 10 7 5 2 0 10 10 6 5 3 9 10 7 1
-------------------------------------------------------------------------------------------

The random numbers for ticket type time are:
-------------------------------------------------------------------------------------------
47 27 35 12 60 98 66 58 81 26 20 14 69 64 14 54 57 88 99 26
-------------------------------------------------------------------------------------------

The random numbers for ticket day time are:
-------------------------------------------------------------------------------------------
80 58 67 18 19 96 74 22 34 85 6 18 56 1 78 62 99 53 11 50
-------------------------------------------------------------------------------------------

The random numbers for number Of tickets time are:
-------------------------------------------------------------------------------------------
92 7 30 48 8 70 50 34 40 56 2 45 5 45 28 40 86 52 50 19
-------------------------------------------------------------------------------------------

The random numbers for casher break time are:
-------------------------------------------------------------------------------------------
62 51 8 78 92 40
-------------------------------------------------------------------------------------------
```

13)Result of Simulation

Result of simulation

n	RN for inter-arrival time	Inter-arrival time	Arrival time	RN for ticket day	Ticket day	RN for ticket type	Ticket type	Number of ticket purchased	Total amount paid
1	0	0	0	42	1	30	1	2	600
2	838	4	4	80	2	49	1	2	600
3	688	3	7	17	1	88	2	2	1600
4	297	2	9	65	2	51	1	3	900
5	450	2	11	95	2	63	1	2	600
6	81	1	12	41	1	7	1	3	900
7	457	2	14	90	2	51	1	2	600
8	554	3	17	99	2	53	1	2	600
9	194	1	18	19	1	66	1	2	600
10	289	2	20	100	2	39	1	2	600
11	150	1	21	49	1	6	1	2	600
12	758	4	25	28	1	51	1	1	300
13	992	4	29	31	1	46	1	3	900
14	36	1	30	74	2	79	2	2	1600
15	865	4	34	56	2	46	1	1	300
16	923	4	38	43	1	40	1	2	600
17	449	2	40	88	2	90	2	2	1600
18	118	1	41	15	1	70	1	3	900
19	763	4	45	37	1	71	1	2	600
20	360	2	47	48	1	33	1	3	900

14) Customer Arrival and Departure Message

```
Counter 1, Counter 2, Counter 3, Counter 4 are in operation.
Counter 4 is exclusively for members only.
Arrival of Customer 1 at minute 0.
Service for Customer 1 starts at minute 0.
Customer 1 bought 2 Day1 Normal tickets from Counter 1.
Arrival of Customer 2 at minute 4.
Service for Customer 2 starts at minute 4.
Customer 2 bought 2 Day2 Normal tickets from Counter 4.
Arrival of Customer 3 at minute 7.
Service for Customer 3 starts at minute 7.
Customer 3 bought 2 Day1 VIP tickets from Counter 1.
Arrival of Customer 4 at minute 9.
Service for Customer 4 starts at minute 9.
Customer 4 bought 3 Day2 Normal tickets from Counter 2.
Arrival of Customer 5 at minute 11.
Service for Customer 5 starts at minute 11.
Customer 5 bought 2 Day2 Normal tickets from Counter 3.
Arrival of Customer 6 at minute 12.
Arrival of Customer 7 at minute 14.
Service for Customer 6 starts at minute 15.
Customer 6 bought 3 Day1 Normal tickets from Counter 3.
Counter 1 takes a 6 minute break at minute 16.
Service for Customer 7 starts at minute 17.
Customer 7 bought 2 Day2 Normal tickets from Counter 2.
Arrival of Customer 8 at minute 17.
Arrival of Customer 9 at minute 18.
Service for Customer 9 starts at minute 18.
Customer 9 bought 2 Day1 Normal tickets from Counter 4.
Service for Customer 8 starts at minute 19.
Customer 8 bought 2 Day2 Normal tickets from Counter 3.
Arrival of Customer 10 at minute 20.
Arrival of Customer 11 at minute 21.
Service for Customer 10 starts at minute 22.
Customer 10 bought 2 Day2 Normal tickets from Counter 1.
Service for Customer 11 starts at minute 23.
Customer 11 bought 2 Day1 Normal tickets from Counter 2.
Arrival of Customer 12 at minute 25.
```

```
Service for Customer 12 starts at minute 25.
Customer 12 bought 1 Day1 Normal tickets from Counter 3.
Arrival of Customer 13 at minute 29.
Arrival of Customer 14 at minute 30.
Service for Customer 14 starts at minute 30.
Customer 14 bought 2 Day2 VIP tickets from Counter 3.
Counter 1 takes a 6 minute break at minute 31.
Service for Customer 13 starts at minute 31.
Customer 13 bought 3 Day1 Normal tickets from Counter 2.
Arrival of Customer 15 at minute 34.
Service for Customer 15 starts at minute 34.
Customer 15 bought 1 Day2 Normal tickets from Counter 4.
Arrival of Customer 16 at minute 38.
Service for Customer 16 starts at minute 38.
Customer 16 bought 2 Day1 Normal tickets from Counter 1.
Arrival of Customer 17 at minute 40.
Service for Customer 17 starts at minute 40.
Customer 17 bought 2 Day2 VIP tickets from Counter 4.
Arrival of Customer 18 at minute 41.
Service for Customer 18 starts at minute 41.
Customer 18 bought 3 Day1 Normal tickets from Counter 2.
Arrival of Customer 19 at minute 45.
Service for Customer 19 starts at minute 47.
Customer 19 bought 2 Day1 Normal tickets from Counter 1.
Arrival of Customer 20 at minute 47.
Service for Customer 20 starts at minute 47.
Customer 20 bought 3 Day1 Normal tickets from Counter 4.
```

15) Simulation Result of Each Counter

Counter 1						
n	RN for service time	Service time	Time service begins	Time service ends	Waiting Time	Time spent in the system
1	8	6	0	6	0	6
3	78	9	7	16	0	9
0	Counter Take A 6 Minutes Break					
10	88	9	22	31	2	11
0	Counter Take A 6 Minutes Break					
16	84	9	38	47	0	9
19	95	9	47	56	2	11

Counter 2						
n	RN for service time	Service time	Time service begins	Time service ends	Waiting Time	Time spent in the system
3	74	8	9	17	0	8
6	30	6	17	23	3	9
11	78	8	23	31	2	10
13	9	4	31	35	2	6
18	86	8	41	49	0	8

Counter 3						
n	RN for service time	Service time	Time service begins	Time service ends	Waiting Time	Time spent in the system
4	67	4	11	15	0	4
5	80	4	15	19	3	7
7	45	3	19	22	2	5
12	65	4	25	29	0	4
14	13	2	30	32	0	2

Counter 4						
n	RN for service time	Service time	Time service begins	Time service ends	Waiting Time	Time spent in the system
0	21	5	4	9	0	5
8	12	4	18	22	0	4
15	40	5	34	39	0	5
17	42	6	40	46	0	6
20	7	4	47	51	0	4

16) Remaining Ticket Table

Remaining Ticket Table		
	Number of Remaining Ticket(s)	
Day	Normal Ticket	VIP Ticket
1	12	13
2	21	11

17) Simulation Result Analysis

```
 FLOW OF CUSTOMER
The average customer interarrival time is 2.350000
The average customer arrival time is 23.100000

 INDIVIDUAL COUNTER PERFORMANCE
 > Counter 1
Average service time is 8.400000
Total sales made is RM  4000.00
Percentage of counter is busy is 0.750000
 > Counter 2
Average service time is 6.800000
Total sales made is RM  3900.00
Percentage of counter is busy is 0.607143
 > Counter 3
Average service time is 3.400000
Total sales made is RM  4000.00
Percentage of counter is busy is 0.303571
 > Counter 4
Average service time is 4.800000
Total sales made is RM  4000.00
Percentage of counter is busy is 0.428571

 OVERALL PERFORMANCE
Average waiting time in queue is 0.000000
Probability that a customer has to wait is 0.000000
Average time spent in system is 6.650000
Percentage of sale for day 1 normal ticket is 65.714286
Percentage of sale for day 1 VIP ticket is 13.333333
Percentage of sale for day 2 normal ticket is 40.000000
Percentage of sale for day 2 VIP ticket is 26.666667
```

6.0 Conclusion

Throughout the project, the team has a deeper understanding on implementing the algorithm design and solving mathematical problems on hand with the simulation of the ticketing system. Also, to make the program more realistic, the team has allowed the counter to have a break while the ticket is being sold.

By the project, the team has explored and practised the skills of using FreeMat to simulate real-time events under controlled conditions.