Contents

Introduction

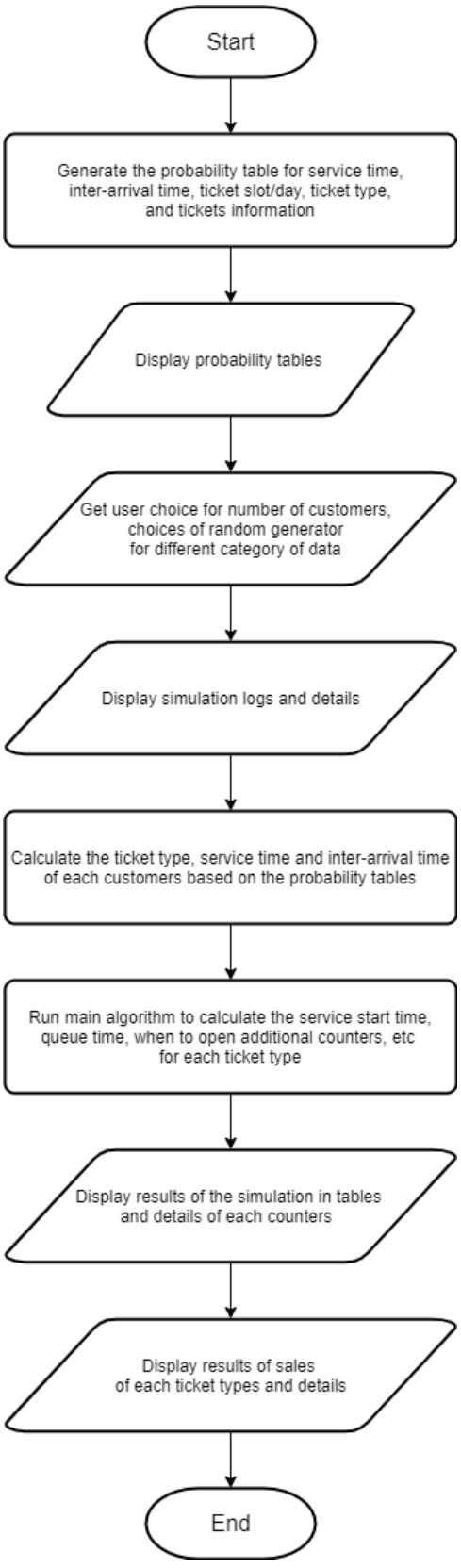
The team has created a queuing simulator for concert ticket purchasing. The main purpose of this simulation is to emulate a concert ticket purchasing scenario whereby every customer is having a good experience in the purchasing process. As the simulator can help to estimate the number of staff needed to serve customers in an efficient and effective way after the gathers of real-time data about service, wait time, and arrival time of customers.

For this queuing system, there are three types of ticket being indicated whereas number of ticket purchased, inter-arrival time, ticket slot/day, ticket type and service time are randomly generated according to the method chosen by user. There are 4 different random number generators being designed for this simulator.

This simulator will display the ticket's information table, the tables of the service time, inter-arrival time, ticket slot/day and ticket type. Then, it will calculate and display the message into activity logs. After that, each table for simulation, counter status and the remaining ticket will be showed. Lastly, details of the waiting time and service time for the counters will also be computed and displayed into the evaluation of results.

Program Logic

A general flowchart based on the main algorithm of the queuing simulator was created as it gives an overall view of what the simulator would accomplish sequentially. This will help everyone to have a better understanding about this simulator. The flowchart is portrayed as below:



FreeMat's built in rand() function generates a floating point number with only 4 decimal places, so the range of possible numbers are from 1 till 10000.Without using a ceil or floor function, the decimals can be removed by multiplying the numbers by 10000.

In order to maximize the flexibility and adjustable part of the simulator, hard-coding any kind of data was minimized with the involvement of complex calculations using multiple variables and modular programming (to organize reusable codes and features into functions).

The subsections hence discuss the logic and reasoning behind the algorithms and methods used in the simulator.

Random Generator

The following are the random generators chosen for this simulator:

- Linear Congruential Generator (LCG)
- Random Variate Generator for Exponential Distribution
- Random Variate Generator for Uniform Distribution
- FreeMat's built-in rand() function

The team chose aforementioned random number generators among all the generators is due to the scope of this project. The team finds that more complicated generators like the Combined Linear Congruential Generator may not be suitable as this project is smaller-scale since Combined LCG is meant for more complex simulations which requires a longer period.

Linear Congruential Generator (LCG)

The LCG requires a, multiplier, c, increment, m, modulus and a seed value. Its purpose is to generate pseudorandom numbers based on a deterministic recursive algorithm, as shown below:

$$X_{n+1} = (aX_n + c) \bmod m$$

The team would randomly generate the a, c and seed values, whereas m is given a value of 10000 as the upper boundary of the random number range for this project is 10000.

Random Variate Generator

Exponential Distribution

The formula for the random variate generator the exponential distribution is:

$$X = \left(\frac{-1}{\boldsymbol{\lambda}}\right) \ln (1 - R)$$

The R value is generated using the rand() function in FreeMat. In order to control the upper boundary and the lower boundary of the generator to produce a value between 1 to 10000, the team has set the λ value to 1, and the output is multiplied with -10000/ (ln 0.0001).
This is because the random function in FreeMat would only produce a value between 0 to 0.9999, and hence the range of X would be from **-ln1 to -ln 0.0001**. Multiplying X with -10000/(ln 0.0001) hence ensure the value would be always between 1 to 10000.

Uniform Distribution

The formula for random variate generator for uniform distribution is:

$$X = a + (b - a)\, R$$

, where R is a random number, which would be generated by the built-in rand() function. Since the range is from 1-10000, a = 0 and b = 10000, and the end result is incremented by 1.

Probability Table

Three types of probability tables need to be generated:

- Probability table for ticket type
- Probability table for service time
- Probability table for inter-arrival time
- Probability table for ticket slot/day

Details of the implementation of the ways to obtain data (eg. validation checking and algorithm for random value generating).

Main Calculation Algorithm

The team has created a calculation algorithm which can do calculations for a particular ticket type. This means, the algorithm needs to be run thrice to fully calculate the list of customers.

The main calculation algorithm needs to:

- Calculate when a customer can be serviced;
- Calculate the additional number of counters that should be opened or closed, based on the number of customers in queue;
- Keep tracks of the customers in queue, and serve them on a first-come-first-serve basis;

The logic flow behind the algorithm is in the following subsection.

Calculation of Service Start Time and Counter Status

The nature of the program allows the user to choose which counters are open in the beginning (the team only takes the number of counters open per ticket type since sequence is trivial). Hence, there are a few possibilities:

- The customer arrives and there is an active counter without people;
- The customer arrives but all active counters are in use;
- The customer arrives, but all active counters are in use and there are no extra counters;

The program determines the current condition for the customer by checking the **last service end time** for each active counter.

Scenario 1: If the arrival time of the customer is more or equal to the last service end time, the customer can then be serviced immediately.

Scenario 2: On the second case, if all the service counters happen to have a service end time greater than the arrival time of the customer, additional counters might be needed. Based on the **requirements to open a new counter (threshold)** which must more than 2 is set by the user, the program determines whether there is a need to open a new counter. It is worth noting that in order to prevent the phenomenon where a counter is open every time there is a person in the queue (it does not reflect real world situation), the program uses another algorithm to check if a counter should be opened. A new counter will open if the number of people in queue is larger than the

threshold whereas if the number of people in queue is less than or equal to 2 persons, the additional counter will be closed.

Scenario 3: If there happens to be no more counter to open when there is a need to, the program simply places the customers in the waiting list, and all counters remain open until the waiting list is reduced to a normal value, which then free counters would be closed.

Calculation of Details of Each Customer

Apart from the generated inter-arrival time, ticket type and service time, there is also a need to calculate the following values:

- Arrival time: Calculated by taking the cumulative inter-arrival time
- Service start time: Calculated by taking the time which is larger than last service end time
- Service end time: Calculated by adding the service start time with service time
- Time in queue: Calculated by taking the service start time (current time) minus the arrival time
- Time in system: Calculated by taking the service end time minus the service start time add the time in queue

At the end of the program, the program would also calculate the mean values of the average inter-arrival time, average arrival time, average service time for all counters, average time in system, average waiting time and probability that a customer has to wait. Sales for each counter will be calculated.

Simulation Logging

The description of what happens at each minute is displayed at the start of the program after all details of the customers' arrival time, service start time, service end time, etc. are collected.

The simulation logging function loops by time (in minute). The logic of the program goes like this, in each of the iteration of the minutes (referred to as t below):

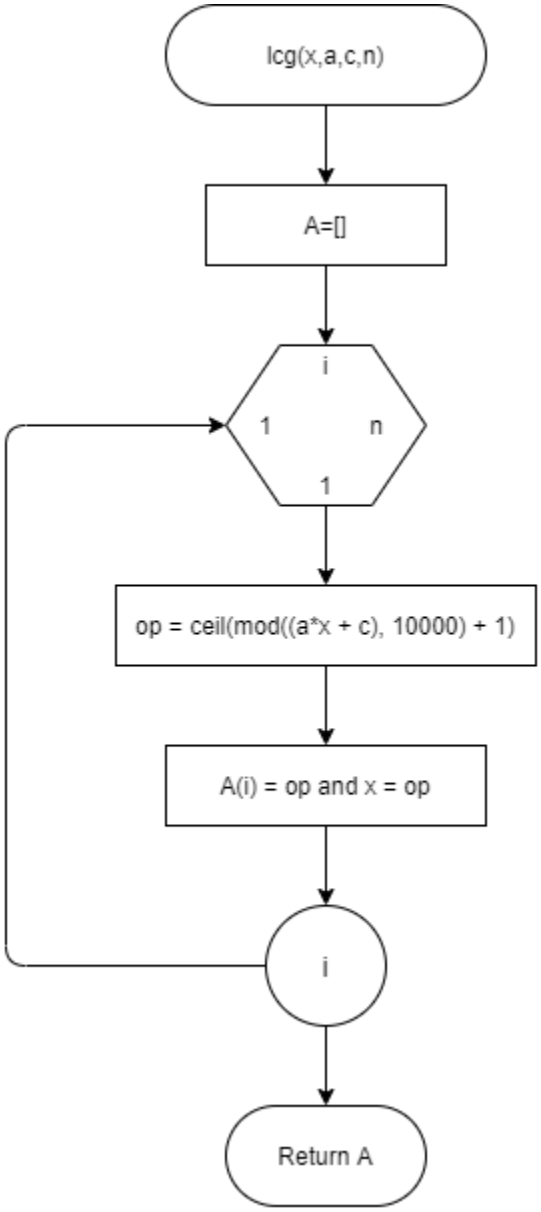
1. Loop through the current time on minute basis
2. If t equals the arrival-time, that means the customer arrives at the time and an arrival message and amount of tickets wish to be purchased are displayed. All the customers are placed into different waiting lists according to the types of ticket
3. Loop through the waiting lists
4. If t is larger and equal to the most recent last service end time, the customers on the waiting list will be serviced on first come first serve basis
5. Add the message that the user is serviced and at which minute
6. Customers which have started services are placed in the list, if the customers are not successfully buy any tickets, their service time will be 0 and add message of it
7. Loop the list
8. If t equals the service end time, announce the customer departs.

*If the customer want to buy 6 tickets, but only 2 tickets are available, a message will be displayed as the customer but 2 tickets instead of 6 tickets.

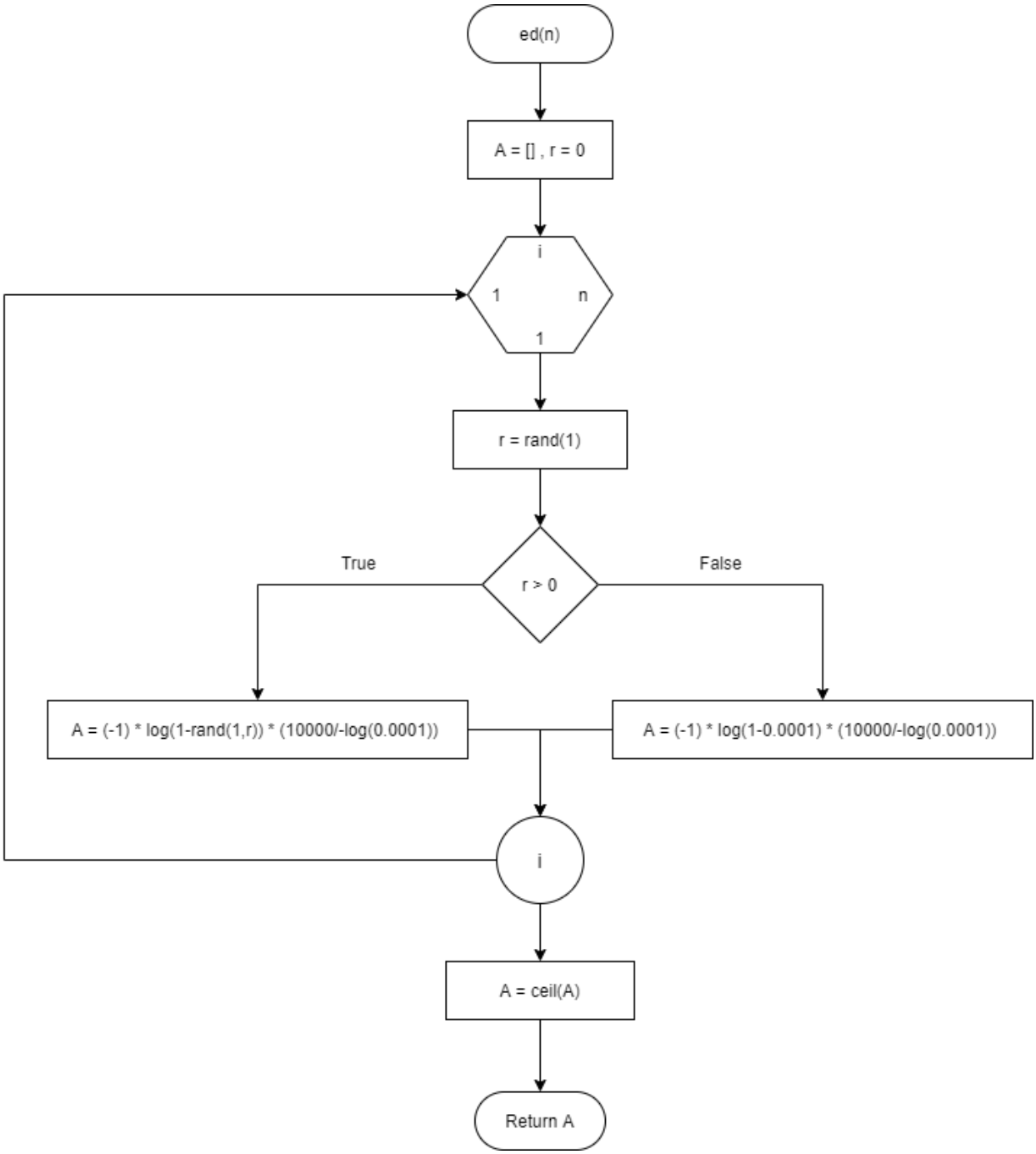
Flowchart

These flowcharts below show the actual arithmetic behind the random number generators.

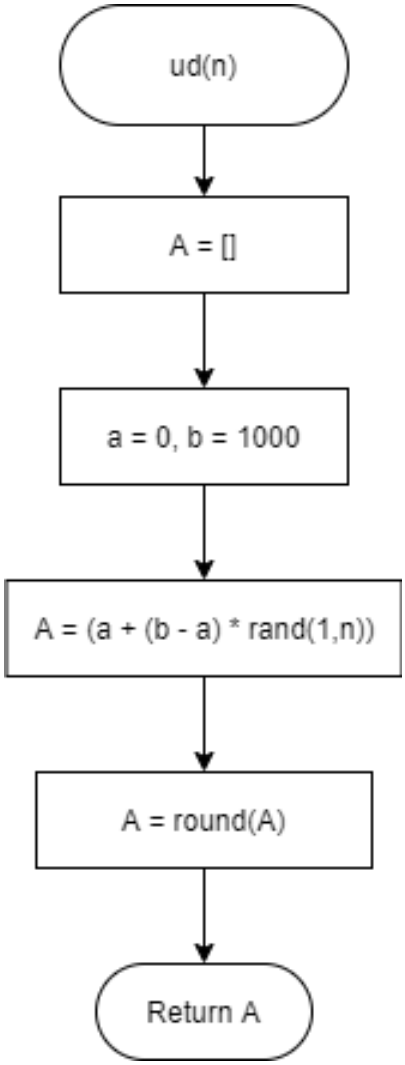
Linear Congruential Generator (lcg.m)



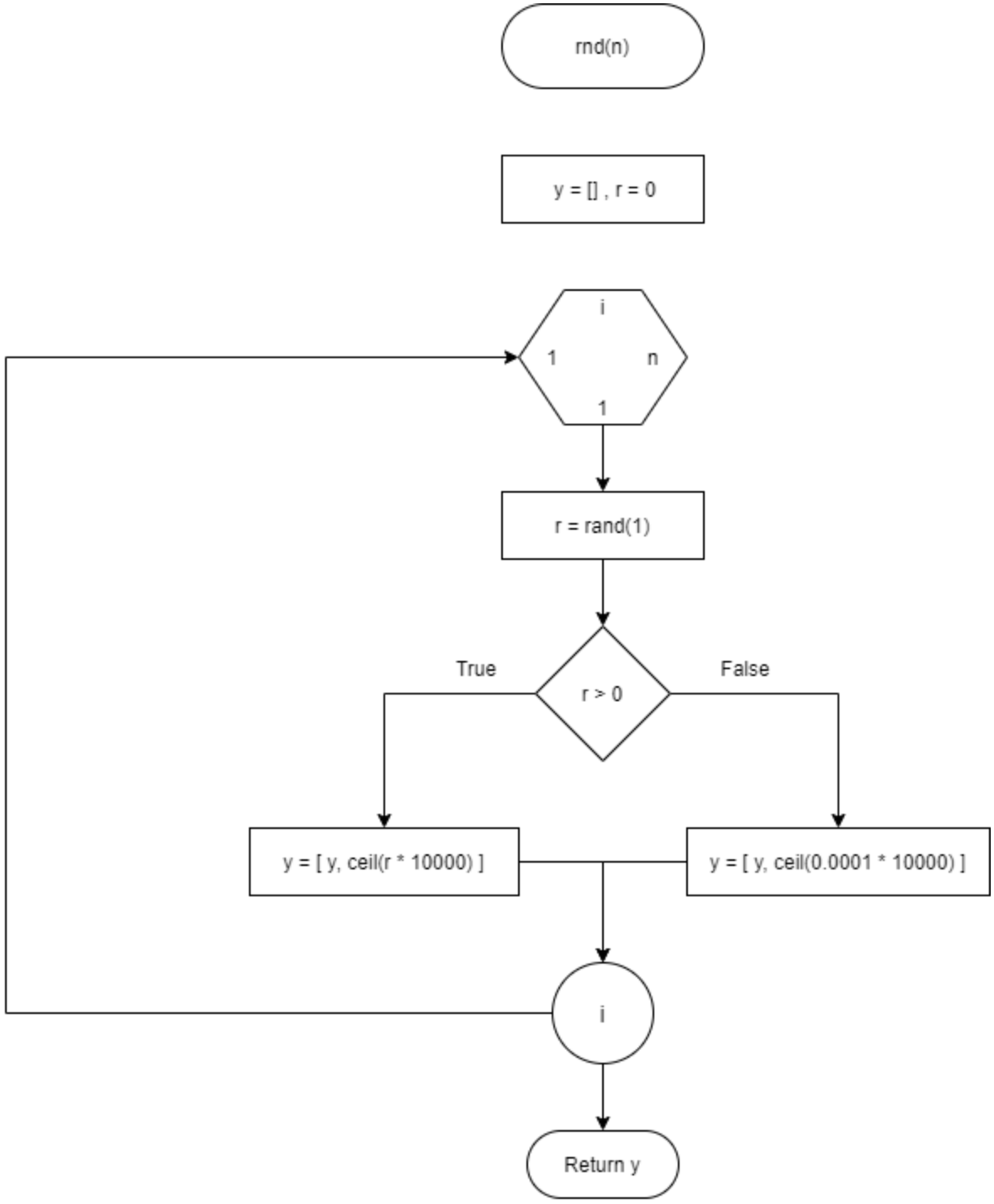
Random Variate Generator for Exponential Distribution (ed.m)



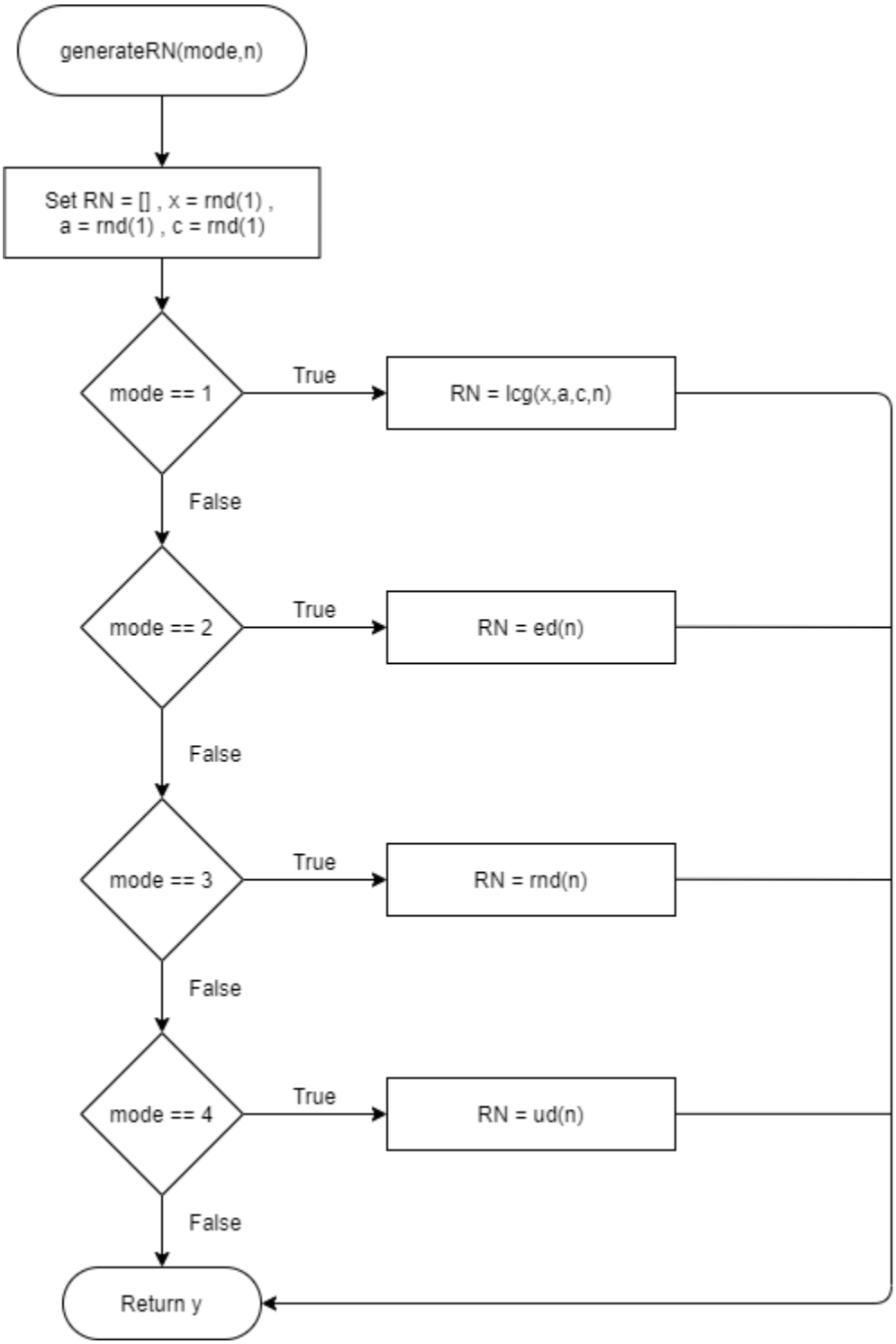
Random Variate Generator for Uniform Distribution (ud.m)



Built-in rand() function (rnd.m)



Generate Random Numbers According to the User Input (generateRN.m)



Sample Simulation

The following documents a sample run out of the infinite possibilities from running the ticket purchasing simulator.

Service Time, Inter-arrival Time, Ticket Slot/Day, Ticket Type and Tickets Information Probability Tables

The simulator begins with displaying the probability tables for service time of each ticket type, inter-arrival time, ticket slot/day, ticket type and tickets information. These probability values were defined in the code beforehand.

```
Ticket type 1: Rockzone Tickets
Ticket type 2: Numbered Seating Tickets
Ticket type 3: Free Seating Tickets
```

Table of Service Time for Ticket Type 1

Service time	Probability	CDF	Range
2	0.2000	0.2000	1 - 2000
3	0.3000	0.5000	2001 - 5000
4	0.3000	0.8000	5001 - 8000
5	0.2000	1.0000	8001 - 10000

Table of Service Time for Ticket Type 2

Service time	Probability	CDF	Range
4	0.1500	0.1500	1 - 1500
5	0.2500	0.4000	1501 - 4000
6	0.2500	0.6500	4001 - 6500
7	0.3500	1.0000	6501 - 10000

Table of Service Time for Ticket Type 3

Service time	Probability	CDF	Range
1	0.1500	0.1500	1 - 1500
2	0.3500	0.5000	1501 - 5000
3	0.3500	0.8500	5001 - 8500
4	0.1500	1.0000	8501 - 10000

Table of Inter-arrival Time			
Inter arrival time	Probability	CDF	Range
1	0.3000	0.3000	1 - 3000
2	0.3500	0.6500	3001 - 6500
3	0.2000	0.8500	6501 - 8500
6	0.1500	1.0000	8501 - 10000

Table of Ticket Slot/Day			
Ticket slot/day	Probability	CDF	Range
1	0.3500	0.3500	1 - 3500
2	0.3000	0.6500	3501 - 6500
3	0.3500	1.0000	6501 - 10000

Table of Ticket Type			
Ticket type	Probability	CDF	Range
1	0.4500	0.4500	1 - 4500
2	0.2500	0.7000	4501 - 7000
3	0.3000	1.0000	7001 - 10000

Table of Tickets Information			
Day/slot	Total ticket type 1	Total ticket type 2	Total ticket type 3
1	3000	2500	1500
2	3000	2500	1500
3	3000	2500	1500

Getting User Inputs

Then the simulator continues with indicating the three ticket types are rockzone tickets, numbered seating tickets and free seating tickets. Then prompting the user to input the number of customers, type of random number generator to use in different aspect. In this case, we set the number of customer as 10 and try to use all four different types of random number generator.

```
Ticket type 1: Rockzone Tickets
Ticket type 2: Numbered Seating Tickets
Ticket type 3: Free Seating Tickets

Please input the number of customer:10

Type of random number generator:
1. Linear Congruential Generator (LCG)
2. Random Variate Generator for Exponential Distribution
3. Random Variate Generator for Uniform Distribution
4. Freemat built-in rand() function
Please choose the type of random number generator for number of ticket purchased: 1
Please choose the type of random number generator for inter-arrival time: 2
Please choose the type of random number generator for ticket slot/day: 3
Please choose the type of random number generator for ticket type: 4
Please choose the type of random number generator for service time: 1
Please set the threshold to open a new counter for ticket type 1(minimum 3): 10
Please set the threshold to open a new counter for ticket type 2(minimum 3): 8
Please set the threshold to open a new counter for ticket type 3(minimum 3): 8
```

Simulation Logs

A simulation log of customer arrival and departure times is displayed with the details of number of tickets purchased on minutes basis. Counter 1,2,3 are for ticket type 1 (rockzone tickets), counter 4,5 are for ticket type 2 (numbered seating tickets) and counter 6,7 are for ticket type 3 (free numbered seating). Operating counters are counter 1,2,4,6 which will not be closed during working hours while additional counters are counter 3,5,7 and will be opened when the thresholds are fulfilled then closed after there no needs for additional counters. Total time of simulation will be calculated in the end.

```
Simulation start.....
Simulation logs:
Counter 1,2,4,6 are operating.
Arrival of customer 1 at minute 1 and 6 tickets were purchased.
Service for customer 1 started at minute 1 at counter 1.
Arrival of customer 2 at minute 2 and 2 tickets were purchased.
Service for customer 2 started at minute 2 at counter 6.
Arrival of customer 3 at minute 3 and 1 tickets were purchased.
Arrival of customer 4 at minute 4 and 5 tickets were purchased.
Service for customer 4 started at minute 4 at counter 2.
Service for customer 3 started at minute 4 at counter 6.
Departure of the customer 2 at minute 4.
Arrival of customer 5 at minute 5 and 2 tickets were purchased.
Service for customer 5 started at minute 5 at counter 1.
Departure of the customer 1 at minute 5.
Arrival of customer 6 at minute 6 and 4 tickets were purchased.
Service for customer 6 started at minute 6 at counter 6.
Departure of the customer 4 at minute 6.
Departure of the customer 3 at minute 6.
Arrival of customer 7 at minute 7 and 2 tickets were purchased.
Service for customer 7 started at minute 7 at counter 2.
Departure of the customer 6 at minute 7.
Arrival of customer 8 at minute 8 and 3 tickets were purchased.
Service for customer 8 started at minute 8 at counter 6.
Departure of the customer 5 at minute 8.
Arrival of customer 9 at minute 10 and 2 tickets were purchased.
Service for customer 9 started at minute 10 at counter 4.
Departure of the customer 7 at minute 10.
Arrival of customer 10 at minute 11 and 2 tickets were purchased.
Service for customer 10 started at minute 11 at counter 1.
Departure of the customer 8 at minute 11.
Departure of the customer 10 at minute 13.

Simulation ends.....
Total time for simulation: 13, meaning that it takes about 0.22 hours for 10 people.
Press enter to continue viewing more details.
```

Result Sets in Tables

The following tables show the details on ticket purchasing based on information of customers, based on different counters for different types of ticket, based on services received by customers and the details of the remaining tickets.

n	RN for Inter-arrival Time	Inter-arrival Time	Arrival Time	RN for Ticket Slot/Day	Ticket Slot/Day	RN for Ticket Type	Ticket Type	Number of Tickets Purchased	Total Amount Paid
1	568	1	1	4863	2	1148	1	6	5400
2	2162	1	2	2523	1	8318	3	2	900
3	1644	1	3	3393	1	9604	3	1	450
4	1902	1	4	1408	1	2966	1	5	4500
5	1316	1	5	6660	3	2350	1	2	1800
6	275	1	6	6898	3	9164	3	4	1800
7	726	1	7	9219	3	3793	1	2	1800
8	1155	1	8	1750	1	9272	3	3	1350
9	3871	2	10	9921	3	6575	2	2	1600
10	190	1	11	2138	1	2392	1	2	1800

The inter-arrival time table and the table of generated customer RN of the tickets and more details on tickets purchased are displayed after generating and calculating data.

Counter 1:

n	RN for Service Time	Service Time	Time Service Begins	Time Service Ends	Waiting Time	Time Spent in the System	Queue No
1	5684	4	1	5	0	4	1001
5	2232	3	5	8	0	3	1003
10	1548	2	11	13	0	2	1005

Average service time for counter 1 = 9/3 = 3.0000
Average time a customer spend in the system for counter 1 = 9/3 = 3.0000
Average waiting time in the queue for counter 1 = 0/3 = 0.0000
Probability that a customer has to wait in the queue for counter 1 = 0/3 = 0.0000
Sales for counter 1 = 10

Counter 2:

n	RN for Service Time	Service Time	Time Service Begins	Time Service Ends	Waiting Time	Time Spent in the System	Queue No
4	873	2	4	6	0	2	1002
7	4769	3	7	10	0	3	1004

Average service time for counter 2 = 5/2 = 2.5000
Average time a customer spend in the system for counter 2 = 5/2 = 2.5000
Average waiting time in the queue for counter 2 = 0/2 = 0.0000
Probability that a customer has to wait in the queue for counter 2 = 0/2 = 0.0000
Sales for counter 2 = 7

Counter 4:

n	RN for Service Time	Service Time	Time Service Begins	Time Service Ends	Waiting Time	Time Spent in the System	Queue No
9	518	4	10	14	0	4	2001

Average service time for counter 4 = 4/1 = 4.0000
Average time a customer spend in the system for counter 4 = 4/1 = 4.0000
Average waiting time in the queue for counter 4 = 0/1 = 0.0000
Probability that a customer has to wait in the queue for counter 4 = 0/1 = 0.0000
Sales for counter 4 = 2

Counter 6:

n	RN for Service Time	Service Time	Time Service Begins	Time Service Ends	Waiting Time	Time Spent in the System	Queue No
2	4116	2	2	4	0	2	3001
3	2750	2	4	6	1	3	3002
6	388	1	6	7	0	1	3003
8	5826	3	8	11	0	3	3004

Average service time for counter 6 = 8/4 = 2.0000
Average time a customer spend in the system for counter 6 = 9/4 = 2.2500
Average waiting time in the queue for counter 6 = 1/4 = 0.2500
Probability that a customer has to wait in the queue for counter 6 = 1/4 = 0.2500
Sales for counter 6 = 10

The result table of each counter that are used in the simulation are calculated and displayed. Counter 1,2,3 are for ticket type 1 (rockzone tickets), counter 4,5,6 are for ticket type 2 (numbered seating tickets) and counter 7,8,9 are for ticket type 3 (free numbered seating), not all counters will be opened during the simulation as there are threshold to be fulfilled in order to open additional counters.

n	Counter	Arrival Time	Start Time	Service Time	End Time	Waiting Time	Time Spent in the System	Ticket Type	Number of Tickets Purchased	Total Amount Paid	Queue No
1	1	1	1	4	5	0	4	1	6	5400	1001
2	6	2	2	2	4	0	2	3	2	900	3001
3	6	3	4	2	6	1	3	3	1	450	3002
4	2	4	4	2	6	0	2	1	5	4500	1002
5	1	5	5	3	8	0	3	1	2	1800	1003
6	6	6	6	1	7	0	1	3	4	1800	3003
7	2	7	7	3	10	0	3	1	2	1800	1004
8	6	8	8	3	11	0	3	3	3	1350	3004
9	4	10	10	4	14	0	4	2	2	1600	2001
10	1	11	11	2	13	0	2	1	2	1800	1005

Each customers' details in purchasing tickets in the system are recorded and calculated then displayed in table form. With this table, a better view of the transaction can be seen.

Day/Slot	Number of Remaining Ticket		
	Ticket Type 1	Ticket Type 2	Ticket Type 3
1	2993	2500	1494
2	2994	2500	1500
3	2996	2498	1496

Total type 1 ticket sold = 17
Total type 2 ticket sold = 2
Total type 3 ticket sold = 10
Total tickets sold = 29

After serving all the customers, the details of the remaining ticket can be generated as above. It showed the quantities of tickets that have been sold and the numbers of remaining tickets that are still available to sell by sorting them with ticket types.

Results of the Simulation

Based on the data and results collected from above, a final result is obtained and derived to turn it into more meaningful and good for decision making in the future for the management.

```
Results of the simulation----------------------------------------------

Average inter-arrival time = 11/10 = 1.1000
Average arrival time = 57/10 = 5.7000
Average service time for all counter = 26/10 = 2.6000
Average time a customer spend in the system for all counter = 27/10 = 2.7000
Average waiting time in the queue = 1/10 = 0.1000
Probability that a customer has to wait in the queue = 1/10 = 0.1000
```

After evaluating the results of simulation, the average inter-arrival time, average arrival time of customer, average service time for all counter, average time a customer spend in the system for all counter, average waiting time in the queue and probability that a customer has to wait in the queue are calculated.

Conclusion

Throughout the entirety of this project, the team has conducted extensive research into the implementation of the program and has learned to dwell systematically into algorithmic designs and representation of an abstract mathematical problem.

At the same time, the team has explored the capabilities and programming methods of FreeMat outside of the subject's syllabus. The team first started formulating the core functions and algorithms needed for the program, which is: the 3 random number generators.

The team came together and extensively shared inputs on devising the algorithm and structure for the main program function and queuing simulator itself.