Implementation

The queuing simulation is simple to use. The model (1) accepts user inputs to (2) run a number of calculations and (3) report on the results. The model is highly flexible, informative, and user friendly. This model will be extremely useful in the analysis and evaluations of queue situations in government service.

Upon opening the model, the user will see the model instructions, located on the "Explanation Tab." The user will follow the simple instruction set.

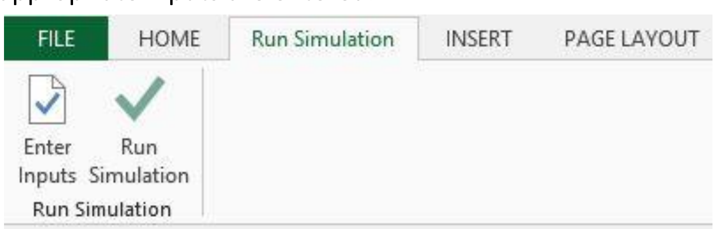
Queuing Simulation

The model is designed to simulate line conditions.

To Use:

1. In the ribbon, select the 2nd tab, "Run Simulation."
2. To enter user inputs, click the button "Enter Inputs."
3. In the yellow cells on the "Report" sheet, enter the conditions you wish to simulate.
4. In the "Run Simulation" tab, click the button "Run simulation."
5. On the "Report" sheet, the green cell identifies how many clients were served.
6. On the "Report" sheet, the orange cells identify the simulation results and statistics.

1. The user will first select the custon tab on the Excel ribbon, entitled "Run Simulation." This tab contains two buttons: "Enter Inputs" and "Run Simulation." The first button will take the user to the "Report" sheet, where the user will enter inputs and see the simulation results. The second button will run the simulation when the appropriate inputs are entered.



2. On the "Reports" sheet, the user will see a various sections and colored cells. The sections at the top of the sheet contain yellow colored cells. These indicate cells that require user inputs.

Inputs		
Time Unit	minutes	
Customer Arrival Rate	5	customers/minutes
Mean Service Time	7	minutes/customer
Number of Servers	3	
Maximum allowed in queue	10	
Simulation run time	200	minutes
Number of Iterations	2	

Time between arrivals (min)	Probaility	Lbound	Ubound
1	0.18	0	0.18
2	0.2	0.18	0.38
3	0.22	0.38	0.6
4	0.25	0.6	0.85
5	0.15	0.85	1

Service Time (mins)	Probability	Lbound	Ubound
4	0.25	0	0.25
5	0.25	0.25	0.5
6	0.25	0.5	0.75

In the **Inputs** section, the first two inputs are actually informed by the two section on the right of the sheet. **Time between Arrivals** and **Service Time** contain statistical distributions of inputs. The inputs *Customer Arrival Rate* and *Mean Service Time* use LOOKUP functions to determine the arrival rate and service time for each customer.

The following inputs will be entered by the user:

   a. *Customer Arrival Rate*The rate at which customers will arrive. Informed by distribution table **Time between Arrivals** located on right.
   b. *Mean Service Time*-The time it takes to serve each customer Informed by distribution table **Service Time** located on right.
   c. *Number of servers*- The number of servers available to assist customers.
   d. *Maximum allowed in queue*-The maximum number of customers allowed in line. One this number is reached, additional customers will be turned away.
   e. *Simulation run time*- The length of time in which servers will assist customers. Customers will cease to arrive once this time has been reached.
   f. *Number of iterations*The number of times the complete simulation will be run. A minimum of 200 iterations is considered best practice.

3. Once the inputs have been entered, the user is ready to run the simulation. On the "Run Simulation" tab of the ribbon, click the button "Run Simulation."

4. Once the simulation has run, the results will be displayed in the orange cells of the "Reports" sheet. Currently, there are 8 outcome cells, indicated by the color orange. These cells require no user input and will be populated by the model. The model outputs are:
   a. *Time last customer leaves*-The time when the last customer completes service and exits the system.
   b. *Average Time in Queue per Customer*-The average length of time a customer spends in line.
   c. *Maximum Time in Queue of all Customers*-The longest amount of time a customer spent in line.
   d. *Average Number of Customers in Queue*-The average number of customers in line at a given time in the simulation.
   e. *Maximum Number in Queue*-The largest size of the line during the simulation.

Time last customer leaves	234.21

Metrics	
Average Time in Queue per Customer	19.06
Maximum Time in Queue of all Customers	36.84
Average Number of Customers in Queue	8.95
Maximum Number in Queue	10

Percentage of Time Servers are Busy	0.97

Number of Customers Processed	110
Number of Customers Turned Away	840
Percentage of Customers Turned Away	0.88

f. *Percentage of Time Servers are Busy*-The rate at which servers are occupied serving customers.

g. *Number of Customers Turned Away*-The amount of customers turned away due to the queue being at maximum capacity.

h. *Percentage of Customers Turned Away*-The percentage of all arriving customers that are turned away.

5. The green cell on the "Reports" sheet indicates the optimized solution to the simulation. In this case, *Number of Customers Processed.*

Number of Customers Processed	110

Assistance

I had a lot of assistance on this project, since it is a project found in the class textbook. I have had a difficult time in this course and wanted to automate a model with which I was already familiar. In my Quantitative Decision Analysis class I created queueing simulation using worksheet functions. I was pleased to see that the textbook contained a queuing simulation in chapter 29. I read the chapter, attempted all parts of the assignment on my own, and then checked my codes against those in the example. I required a lot of clarification and correction from the textbook. As such my model is nearly identical to that found in the book. I then attempted the exercises at the end of the chapter (which contain no instructions). I did not do very well on these, though I have left in the model m code to document my attempts.

Learning and Conceptual Difficulties

This class has been difficult for me, so this project was quite challenging for me; however, it also clarified several concepts for me. First, the project clarified the idea of event-based programming. I was initially confused by the concept, but have found that it is actually quite simple. Each event is scheduled to assign values to certain variables. Events often depend on counter variables, on which this model relies greatly. The counters necessary to record the number of arriving customers, the size of the line, and the total amount of time that has passed in the simulation. The counters are crucial to the execution of each step.

I also solidified my understanding of the diverse type of variables and arrays. This model relies on a large amount of globally declared variables, dimensioned as arrays, strings, single, integer, and variant. This project helped me understand when each of these variables is appropriate. I also encountered the difficulty of global variables. Globally declared variables affect any sub-procedure in which they take part. This concept gave me difficulty as I attempted to keep the events of my sub-procedures in the proper order.

This project gave me a great deal of trouble. I struggled to keep my loops in proper order and function. Since the model is event based, each event must be repeated several times. In this aspect I relied heavily on the textbook to correct me and keep my loops organized. One aspect I wanted to include was a Monte Carlo simulation of the model. This would accept the **Iteration** user input and repeat the entire simulation that according to the input. The statistical results of each iteration would then be recorded on a summary sheet. I was able to input a simple Do loop to repeat the simulation the specified number of times, but was unable to update the summary sheet as desired. I could not get the model to preserve the outputs of the previous iterations. Instead, the model would take the most recent outputs and place those in the results of *each* iteration. I spent many hours trying to correct this by means of different methods, including locally declared variables, various loop types, and the placement of the code. In the end I was unable to correct this procedure before the assignment deadline.

Another area that gave me difficulty is simulating the number of arrivals that determine the line is too long and turn away, or balk. To attempt this, I declared new variables: the balk rate (or percentage of customers who may balk) and the required line length that triggers arrivals to balk. While I could calculate this, I found that it wasn't executing correctly. The code caused far too many arrivals to balk. I could not figure out why this occurred and commented out the relevant code.