This program has been made to speed up typical queue theory calculations. It's able to calculate important data for the most common cases. All notation (and the formulas used by the calculations) is exactly the same as in the book "Introduction to Operations Research", Hillier-Lieberman, McGraw-Hill Inc., which is the book used in my Operations Research class.
It you have any questions, please read chapter 15 of the named book.

All systems are supposed to follow exponential distributions for service times,(which is equivalent to say that all systems follow Poisson distributions for arrival rates).

This program calculates Poisson input and exponential service times models. It returns the most important values for each model (queue length, service times, etc…).

Terminology and Notation

The following standard terminology and notation will be used:

State of system = number of customers in queueing system.

Queue length = number of customers waiting for service

= state of system minus number of customers being served.

N(t) = number of customers in queueing system at time t ($t \geq 0$).

Pn(t) = probability of exactly n customers in queueing system at time given number at time 0.

s = number of servers (parallel service channels) in queueing system.

λn = mean arrival rate (expected number of arrivals per unit time) of new customers when n customers are in system.

μn = mean service rate for overall system (expected number of customers completing service per unit time) when n customers are in system. Note: μn represents combined rate at which all busy servers (those serving customers) achieve service completions.

ρ, μ, λ = see following paragraph.

When λn is a constant for all n, this constant is denoted by λ. When the mean service rate per busy server is a constant for all $n \geq 1$, this constant is denoted by μ. (In this case, $\lambda n = s\mu$ when $n \geq s$, that is, when all s servers are busy.) Under these circumstances, $1/\lambda$ and $1/\mu$, are the expected interarrival time and the expected service time, respectively. Also, $\rho = \lambda/s\mu$ is the utilization factor for the service facility ,i.e., the expected fraction of time the individual servers are busy, because $\lambda/(s\mu)$ represents the fraction of the system's service capacity ($s\mu$) that is being utilized on the average by arriving customers (λ).
Certain notation also is required to describe steady-state results. When a queueing system has recently begun operation, the state of the system (number of customers in the system) will be greatly affected by the initial state and by the time that has since elapsed. The system is said to be in a transient condition. However, after sufficient time has elapsed, the state of the system becomes essentially independent of the initial state and the elapsed time (except under unusual circumstances). The system has now essentially reached a steady-state condition, where the probability distribution of the state of the system remains the same (the steady-state or stationary distribution) over time. Queueing theory has tended to focus largely on the steady-state condition, partielly because the transient case is more difficult analytically. (Some transient results exist, but they are generally beyond the technical scope of this book.) The following notation assumes that the system is in a steady-state condition:

Pn = probability of exactly n customers in queueing system.

L = expected number of customers in queueing system.

Lq = expected queue length (excludes customers being served).

$\underline{W}$ = waiting time in system (includes service time) for each individual customer.

W = E($\underline{W}$).

Wq = waiting time in queue (excludes service time) for each individual customer.

Relationships between L, W, Lq, and Wq

Assume that λn is a constant λ for all n. It has been proved that in a steady-state queueing process,

$$L = \lambda \cdot W$$

the same proof also shows that

$$Lq = \lambda \cdot Wq$$

If the λn are not equal, then λ can be replaced in these equations by λ_, the average arrival rate over the long run. (We shall show later how λ_ can be determined tome basic cases.)

Now assume that the mean service time is a constant, 1/μ, for all n ≥ 1. It then follows that

$$W = Wq + \frac{1}{\mu}$$

These relationships are extremely important because they enable all four of the fundamental quantities-L, W, Lq, and Wq-to be immediately determined as soon as one is found analytically. This situation is fortunate because some of these quantities are much easier to find than others when a queueing model is solved from basic principles.

Use:
Choose Queue, that is the program that calls the other subprogram. QUEUE is made to choose between the different model cases considered. These ones are:

MMS:
The most basic and typical one. This model is the special case of the birth-and-death process where the queueing system's mean arrival rate and mean service rate per busy server are constant (λ and μ,respectively), regardless of the state of the system.

Input:
A new window is showed. It prompts for some required values, (except n, which is optional).

λ= Average arrival rate

μ= Average service rate

s= number of servers

ρ= utilization factor: remember that it must be pre-calculated by hand using: $\rho = \lambda/s\mu$

Note that, when ρ is required, it must be entered. ρ is $\rho = \lambda/s\mu$, and you may see that wherever λ ,μ and s are prompted, ρ is still prompted. This means that you should calculate $\rho = \lambda/s\mu$ by hand. Why this when the calc has al the values required to calculate ρ itself ? This is just because I forgot to program such an easy calculation, when I realized of that the program was ready and working, and besides, It's an easy calculation, so do it by hand (I'm a lousy programmer)

n= number of probabilities that will be calculated (optional). The program will calculate the probabilities of n customers in the queueing system.
It's optional and, though it may say n=0 as the standard value,it calculates probabilities of up to 4 customers in system (if no greater-than-4 value is entered).

Output:
6: Po: probability of exactly 0 customers (no customers) in queueing system

5: Lq= length of the queue (expected queue length,excludes customers being served)

4: Wq= waiting time in queue

3: W= waiting time IN SYSTEM

2: L= expected number of customers in queueing SYSTEM

1: Pn = array of 1x(n+1) elements, which has the form: (P0,P1,P2…Pn), where each Pn is the probability of exactly n customers in queueing system. This way, obviously, the first array element is the "P0" that I have explained just a couple of lines ago. Note that if during the input, the "n" value wasn't entered, the program calculates still the array for n=4 .

MMS / C: finite queue

The number of customers in the system is not
permitted to exceed some specified number. Any customer
that arrives while the queue is "full" is refused to
enter the system and so leaves forever.

Input:
A new window is showed. It prompts for some required values.

λ= Average arrival rate

μ= Average service rate

s= number of servers

ρ= utilization factor: remember that it must be pre-calculated by hand using: $\rho = \lambda/s\mu$

C= queue size limit. When the queue size is C, (that is, when "C" customers are waiting in queue),Any new customer entering the system will decide not to do it and so will leave the system.

Output:

7: Po: probability of exactly 0 customers (no customers) in queueing system

6: Lq= length of the queue (expected queue length,excludes customers being served)

5: Pn = array of 1x(n+1) elements, which has the form: (P0,P1,P2…Pn), where each Pn is the probability of exactly n customers in queueing system. This way, obviously, the first array element is the "P0" that I have explained just a couple of lines ago.Note that for this case, n=C, thas is, all probability cases are computed. C is the queue limit, and so , the array has (c+1) elements. Obviously, if C is the queue limit, it doesn't make sense to calculate probability of more than C customers in queue.

4: L= expected number of customers in queueing SYSTEM

3: λ2= this value is used in the Hillier-Lieberman book , the book uses it in the formulas that this program uses to calculate everything. I have included this value (useless value, indeed), because in my class I'm supposed to use it when calculations are made by hand. I just didn't want to make it so clear that I was using a HP49. There is no meaning (I think) for this λ2 value, but is equal to :

$$\lambda 2 \equiv \overline{\lambda} = \sum_{n=0}^{\infty} \lambda_n Pn = \lambda(1 - Pc)$$

2: W= waiting time IN SYSTEM

1: Wq= waiting time in queue

MMS/priority:

All the models so far have assumed that the arrival rate is always a constant this won't be the case now. Now, different classes are considered. Every one of them has a different value for λ.
Why is this? There are many systems in which customers don't belong to a single class, that is, different kinds of customers are considered. This means that , for a special class of customer, there may be a special (faster)queue. For example: in airports, airlines usually have a different counter for business-class customers, which allows them to check in faster than average customers.

The model considered here is a nonpreemptive priority model, this means that when a customer is being served, this service will continue until its end, even if a higher-priority customer enters the system. In other words, a customer being served cannot be ejected back into the queue (pre-empted) if a higher priority customer enters the queueing system.

Input:

λ= arrival rates VECTOR . This must be a vector of the form: (λ1..λn) , where every λ is the arrival rate for every considered class . In this type of queue model, where different arrival rates are considered, customers are divided in classes. Every one of them has a λ value . Note that, although there are different arrival rates, there is just one service rate.

μ= Average service rate. The same for all considered customer classes.

S= number of servers.

Lq= MMS equivalent queue length. This value is the Lq of the equivalent MMS queue model. This is made just calculating the Lq for the MMS model in which the λ is computed as the sum of all the λ_n, that is:

$$\lambda = \sum_{i=1}^{n} \lambda_i$$ for "i" considered classes.

This means that, before running these option of the program (MMS / dependent rates), you must run the MMS option (the first one in the menu) and calculate Lq, for a system with a $\lambda=\Sigma\lambda_i$

Output:

8: A

7: ValoresB = This values, "A", and "valoresB" are just numeric values used in the formulas needed to calculate all important data. The program outputs these values just in case the user (you) wants to know these values. That is because , as I have explained earlier, I don't want my teacher to know that I use the HP49 to calculate everything. These "A" and "valoresB" are just intermediate values that have no special meaning. You won't need them.

6: λ
5: ρ----------------------------→(utilization factor Vector for all classes)
4: Lq
3: Wq
2: W
1: L --> All these six values are the same values as in the MMS system. The only difference is that now they are vectors. Each value in the vector is the value of the considered class. Suppose there are 2 different classes, then all these values, λ ,ρ , Lq, Wq, W, L, are vectors of size 2. For example, in Lq(k1, k2). k1 would be the queue length for the first considered class.

MMS/Finite population:

The input source is limited. Let N denote the size of the population. Thus, when the number of customers in the queueing system is n, there are only N-n potential customers remaining in the input source.

Input:

λ = average arrival rate

μ = average service rate

s= number of servers

n= population size

output:

7: Po= probability of exactly 0 customers (no customers) in queueing system

6: Lq= length of the queue (expected queue length,excludes customers being served)

5: Pn = array of 1x(n+1) elements, which has the form: (P0,P1,P2…Pn), where each Pn is the probability of exactly n customers in queueing system. This way, obviously, the first array element is the "P0" that I have explained just a couple of lines ago. The program will calculate Pn for n=N, that is, for all the population.

4: L= expected number of customers in queueing SYSTEM

3: λ2= this is a value used in the formulas. It has no special meaning. The program outputs this value just in case you want to check calculations by hand. This value equals:

$$\overline{\lambda} = \sum_{n=0}^{\infty} \lambda_n P_n = \lambda \cdot (N - L)$$

2: W= waiting time IN SYSTEM

1: Wq= waiting time in queue

The program is written in USER-RPL. The main program is QUEUE. The other ones are just the sub-program that make all calculations and prompt the needed values. You may use them without running QUEUE, so, if you need to make calculations for a MMS system, you may run MMS directly instead of running first QUEUE and choosing MMS.
I have always used the program in RPN mode.

IMPORTANT: the Calc-CAS must be set in Not-numeric mode. This means that in the CAS menu, "numeric " must not be checked. In other words, system-flag 03 must be cleared.

SETUP: all files must be copied in the same directory. In order to transfer the files to the calc, Kermit mode must be used, and Translation mode 3 must be set. Note that in the files, special Hp characters are used and so, translation Chr 128-255 must be set.
In order to do this, just run the Pc connectivity kit program. Then in "calculator" choose "communication settings" ("comm Settings") and in "translation" choose mode 3. This should work properly. But if this is not the case, then while the calc is on "server mode", press in the calc ON, then choose APPS (button "G " in the HP49) and choose "2. I/O functions", then "5. Transfer" , after this make sure "Kermit " mode is chosen in "type", and, in "Xlat" choose Chr 128-255. After this click on F4. Now, in the PC (in the PC connectivity kit program) choose the files to transfer and do it like always (drag and drop). This should work always, if not, try to do it one-file-at-a-time.

Files that must be transferred:

QUEUE
MMS
MMSC
MMSN
MMSPRIOR

I’m not responsible for any damage that this program may cause to your calc. However, it’s a USER RPL program , so it’s not dangerous at all. I have it tested on HP49 ROM 1.19-6

Manuel Peral Pardo   manuperal@gmail.com
Ingeniería Industrial
EPS Ferrol
Universidad de La Coruña
SPAIN