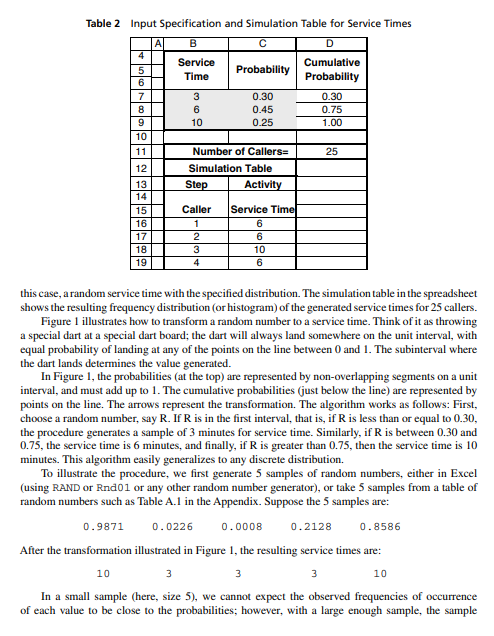
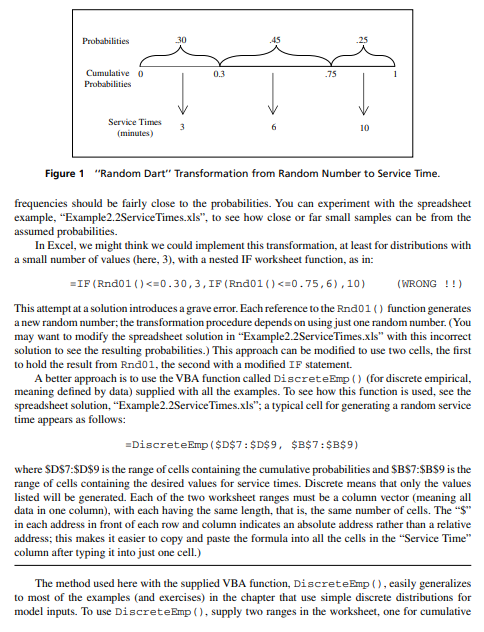
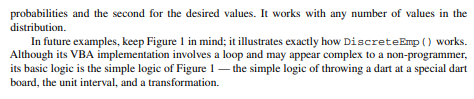
* Use Random Numbers to Generate Arrival and Service Time with an example

Example 2: Random Service Times An automated telephone information service spends either 3, 6, or 10 minutes with each caller. The proportion of calls for each service length is 30%, 45%, and 25%, respectively. We want to simulate these service times in a spreadsheet. Our actual purpose is to learn how to generate random samples from a discrete distribution, in preparation for the queueing, inventory, and other examples to follow. This also is an example of a Monte Carlo simulation. Table 2 shows the input specification and a portion of the simulation table (the first 4 callers), taken from the spreadsheet model, “Example2.2ServiceTimes.xls”. The input specification defines the probabilities for the service times, and also includes the cumulative probabilities. Cumulative probabilities always increase to 1.0; as we shall see, the generation method uses the cumulative probabilities. The method always starts with a random number and ends with the desired value, in

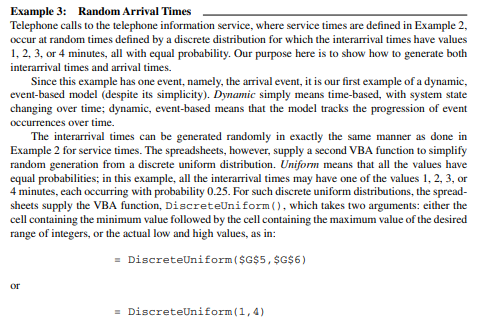


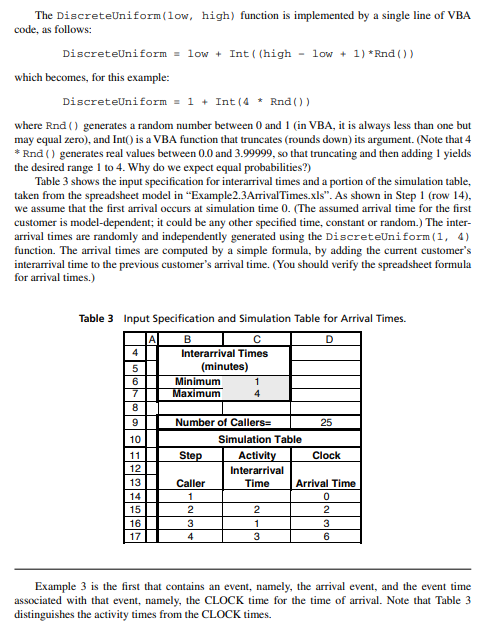




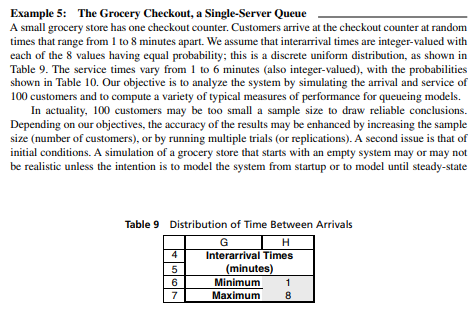
* Monte Carlo Simulation single server queuing model having Inter-arrival and Service times Constant

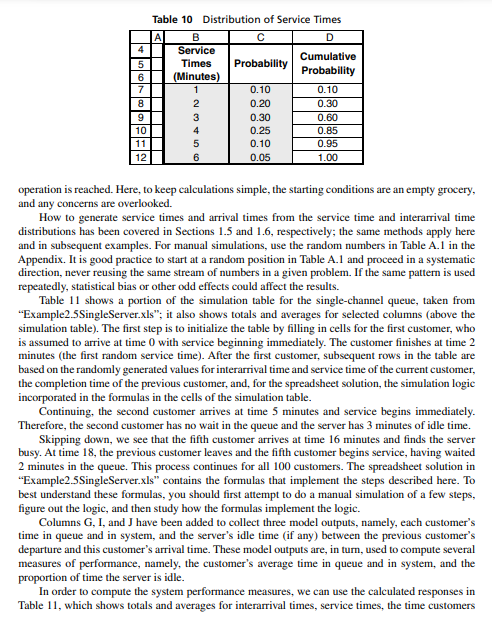
Example 3: Random Arrival Times Telephone calls to the telephone information service, where service times are defined in Example 2, occur at random times defined by a discrete distribution for which the interarrival times have values 1, 2, 3, or 4 minutes, all with equal probability. Our purpose here is to show how to generate both interarrival times and arrival times. Since this example has one event, namely, the arrival event, it is our first example of a dynamic, event-based model (despite its simplicity). Dynamic simply means time-based, with system state changing over time; dynamic, event-based means that the model tracks the progression of event occurrences over time. The interarrival times can be generated randomly

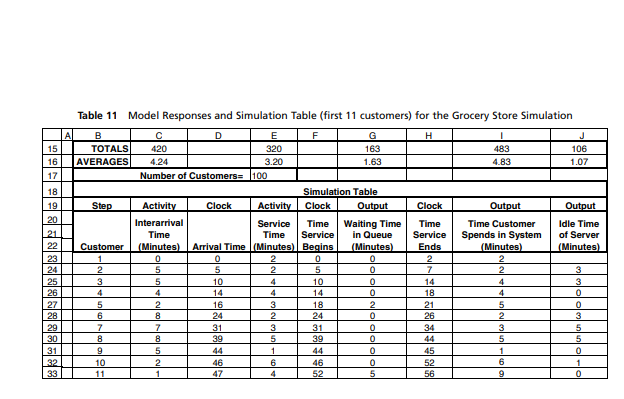


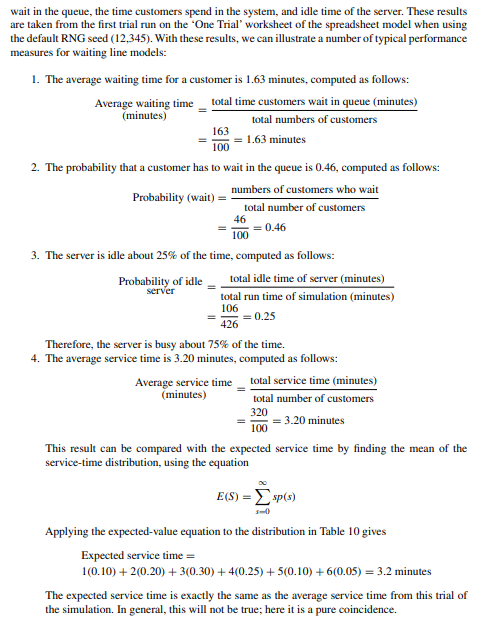


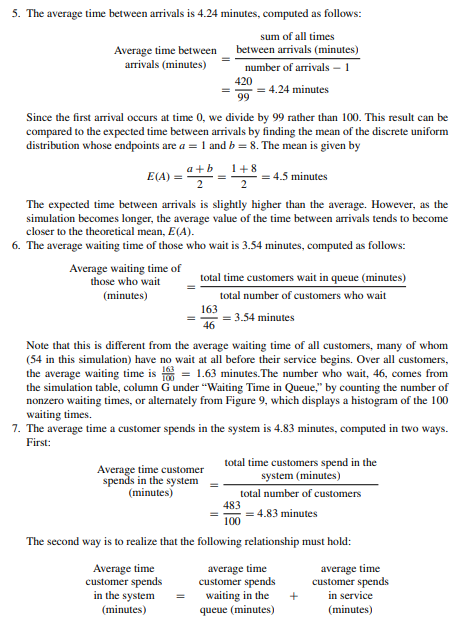
Example 5: The Grocery Checkout, a Single-Server Queue A small grocery store has one checkout counter. Customers arrive at the checkout counter at random times that range from 1 to 8 minutes apart. We assume that interarrival times are integer-valued with each of the 8 values having equal probability; this is a discrete uniform distribution, as shown in Table 9. The service times vary from 1 to 6 minutes (also integer-valued), with the probabilities shown in Table 10. Our objective is to analyze the system by simulating the arrival and service of 100 customers and to compute a variety of typical measures of performance for queueing models.

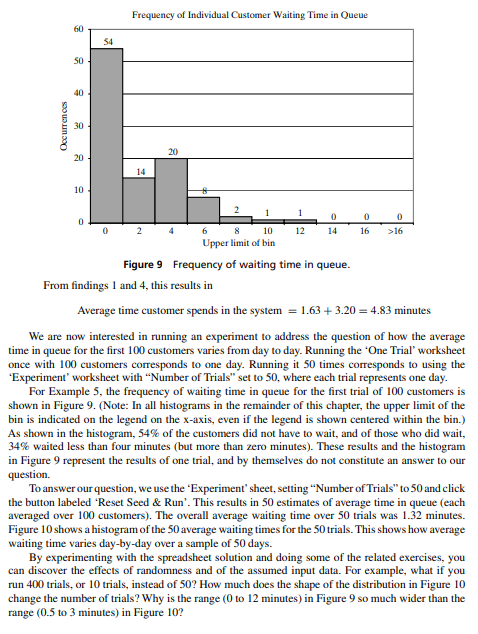


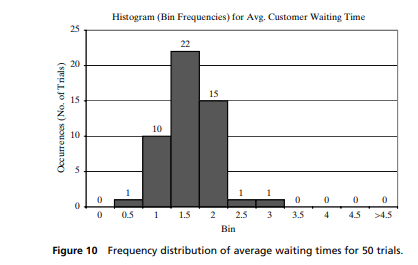








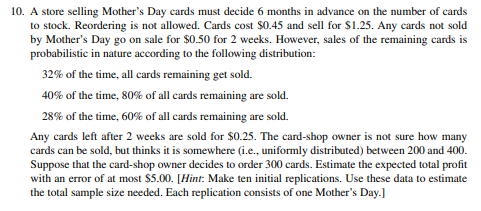




* Monte Carlo simulation technique, determine the average profit of a company

Q10, Chapter 11. Estimation of absolute performance, page 415

A store selling Mother’s Day cards must decide 6 months in advance on the number of cards to stock. Reordering is not allowed. Cards cost $0.45 and sell for $1.25. Any cards not sold by Mother’s Day go on sale for $0.50 for 2 weeks. However, sales of the remaining cards is probabilistic in nature according to the following distribution: 32% of the time, all cards remaining get sold. 40% of the time, 80% of all cards remaining are sold. 28% of the time, 60% of all cards remaining are sold. Any cards left after 2 weeks are sold for $0.25. The card-shop owner is not sure how many cards can be sold, but thinks it is somewhere (i.e., uniformly distributed) between 200 and 400. Suppose that the card-shop owner decides to order 300 cards. Estimate the expected total profit with an error of at most $5.00. [Hint: Make ten initial replications. Use these data to estimate the total sample size needed. Each replication consists of one Mother’s Day.]



Example 7: The News Dealer’s Problem A news dealer buys papers for 33 cents each and sells them for 50 cents each. Newspapers not sold at the end of the day are sold as scrap for 5 cents each. Newspapers can be purchased in bundles of 10. Thus, the newsstand can buy 50 or 60 or 70 papers, and so on. The order quantity, Q, is the only policy decision. Unlike some inventory problems, the order quantity Q is fixed since ending inventory is always zero due to scrapping leftover papers.

Solution:

