# rMcRonald's Restaurant

# (Original lab from CS 514, Washington University of St. Louis, Summer 2001.)

# McRonald.java (1 queue and 1 service window)

McRonald's fast food restaurant may not have great food but it prides itself being faster than its major competitors: Burger Thing, Hardlee's, and Taco Snail. McRonald's requires between 2 and 7 minutes to fill a customer order. On average, a customer arrives at McRonald's every 5 minutes. The manager would like for you to perform a *simulation* in order to gather statistics about the amount of time customers have to wait between the time they arrive in the store and the time they get their food.

This simulation is not yet well-defined. Part of your job is to define it more precisely. What information would be useful to the manager? Total served, average wait time, longest wait time, longest queue

What data do you have to track? Time, number of customers in queue

How many queues are at this McRonald's? 1

How many service windows are at this McRonald's? 1

How many hours is it open each day? 18 hours – 6 am to 12 am How many minutes? 1080 minutes

What happens to customers who arrive at 11:59 pm but are not yet served by midnight? Serve them all

Hint: The fact that one customer arrives every 5 minutes means that there is a 20% chance that a customer will arrive in any given minute. Some of you will need to refresh your memory of Math.random.

**Design Considerations**

Do you need a customer class? If so, what private data should it store?

If not, what data do you store in the queue?

What happens in each minute? What might happen in each minute? What happens at 12:00am (midnight)?

-- display the minute and queue.toString(), e.g., perhaps at the 60th minute, 59: [56, 58]

--

--

--

--

--

Write a pseudocode algorithm for the McRonald simulation.

**McRonald.java**

The shells for each McRonald simulation set up the variables, the accessor methods, and the output file. Every minute, the simulation sends the customer queue to the output file and counts down the customer being served. At the end, the simulation prints the statistics to the screen.

|  |  |
| --- | --- |
| Customer queue (partial) to file: | Statistics to the screen: |
| . . .  55: [49, 50, 52, 53]  49 is now being served for 3 minutes. 56: [49, 50, 52, 53]  49 is now being served for 2 minutes. 57: [49, 50, 52, 53]  49 is now being served for 1 minutes. 58: [50, 52, 53, 58]  50 is now being served for 3 minutes. 59: [50, 52, 53, 58]  50 is now being served for 2 minutes. 60: [50, 52, 53, 58, 60]  50 is now being served for 1 minutes. 61: [52, 53, 58, 60, 61]  52 is now being served for 6 minutes.  . . . | 1 queue, 1 service window, probability of arrival = 0.2 Total customers served = 219 Average wait time = 16.689497716894977 Longest wait time = 47 Longest queue = 13 |

**McRonald1000.java**

Is one day's data reliable enough? Probably not. Copy McRonald.java into McRonald1000.java. Modify it to run it for 1000 days and display the statistics, including the average number of customers served per day and the largest number served per day. There is no need to output the queue to a file.

|  |  |
| --- | --- |
| No file. | Statistics to the screen: |
|  | 100 days, 1 queue, 1 service window, probability of arrival = 0.2  Total customers served = 215735  Average wait time = 74.58683106589102  Longest wait time = 365  Longest queue = 64  Average served per day = 215.735  Largest day = 260 |

# McRonald3.java (1 queue and 3 service windows)

What happens if the restaurant adds more service windows? Copy McRonald.java into McRonald3.java and modify it to use 3 service windows. The service windows can be modeled by an array or by its own ServiceWindow class.

|  |  |
| --- | --- |
| Customer queue (partial) to file: | Statistics to the screen: |
|  | 1 queue, 3 service windows, probability of arrival = 0.5  Total customers served = 559 Average wait time = 13.101967799642217 Longest wait time = 36 Longest queue = 18 |

# 

# McRonaldVIP.java (2 queues and 1 service window)

Every now and then a VIP shows up at McRonald's. He or she always goes straight to the VIP queue, which gets served before the regular queues. Copy McRonald.java into McRonaldVIP.java and modify it to allow for the random (and rare) arrival of a VIP. If the VIP queue has a customer, serve him/her before the people in the regular queue. Output both queues to the same file. Report the four statistics for the normal customers, the number of VIP's, and the average VIP wait time.

|  |  |
| --- | --- |
| Customer queue (partial) to file: | Statistics to the screen: |
| . . .  100: [96, 98] VIP 100: []  96 is now being served for 2 minutes.  101: [96, 98] VIP 101: []  96 is now being served for 1 minutes.  102: [98] VIP 102: [102] VIP 102 is now being served for 5 minutes.  103: [98] VIP 103: [102] VIP 102 is now being served for 4 minutes.  104: [98] VIP 104: [102] VIP 102 is now being served for 3 minutes.  105: [98] VIP 105: [102] VIP 102 is now being served for 2 minutes.  106: [98] VIP 106: [102] VIP 102 is now being served for 1 minutes.  107: [98] VIP 107: []  98 is now being served for 6 minutes.  108: [98] VIP 108: []  98 is now being served for 5 minutes.  109: [98] VIP 109: []  . . . | 1 queue plus VIP queue, 1 service window probability of arrival = 0.2 probability of VIP = 0.02 Total customers served = 432 Total VIPs served = 20 Average wait time = 5.469907407407407 Average VIP wait time = 11.85 Longest wait time = 32 Longest queue = 7 |

Regular customer who arrived at minute 96 was being served. Regular customer who arrived at minute 98 got in line. A VIP arrived at minute 102. The VIP got served first. Then regular customer 96 was served.

# McRonald5.java (surges in arrival times)

There are surges of customers at breakfast, lunch, and dinner, and corresponding down times between these rushes. Copy McRonald.java into McRonald5.java and use a sinusoidal equation to model more realistic customer arrival rates based on time of day. This time, enqueue the actual arrival times, e.g., 6:08: [6:02, 6:05, 6:08]. Look at the queue in the file to see the surges at breakfast, lunch, and dinner.