Caleb Pudvar

Producer/Consumer Analysis

Although my solution to the producer/consumer problem has some quirks to it, I believe it sufficiently demonstrates the properties this problem should have while simultaneously having no deadlock. In order to reach deadlock in this problem, both the producer and consumer would have to find themselves asleep at the same time. In order to get around this ever being an option, my solution simply has the consuming thread wait for a given time if there is no quantity left to consume. Since the wait() either waits for a notify() OR waits for X amount of ms, the thread will at worst wait for X time before cycling through the thread’s process again. This same action is taken with the producing thread. If the buffer is full, it simply waits for a set amount of time to allow for the producer to make room. When a quantity is produced, it also notifies the consumption thread that there is more to consume. In this manner, there is no way for both threads to simultaneously sleep, therefore essentially preventing deadlock altogether.

On the following page will show my output for a few different cases. Please note I have built my solution around a restaurant theme for ease of readability. Because of this:

* Buffer size: food on plate
* Producer: meal being delivered
* Consumer: total meals consumed

For brevity, buffer size = 3 and total to consume = 10 (instead of buffer = 50 and consume = 1000)

|  |  |  |
| --- | --- | --- |
| **Standard**  (producer executed first) | **Consumer First**  (reverse thread.start() for producer and consumer) | **For Buffer = # to Consume**  (Buffer, # to Consume = 5) |
| Delivering Meal #1  Total amount of food on plate: 1  Delivering Meal #2  Total amount of food on plate: 2  Total Meals Consumed: 1  Total Meals Consumed: 2  Delivering Meal #3  Total amount of food on plate: 1  Total Meals Consumed: 3  Delivering Meal #4  Total amount of food on plate: 1  Total Meals Consumed: 4  Delivering Meal #5  Total amount of food on plate: 1  Total Meals Consumed: 5  Delivering Meal #6  Total amount of food on plate: 1  Total Meals Consumed: 6  Delivering Meal #7  Total amount of food on plate: 1  Total Meals Consumed: 7  Delivering Meal #8  Total amount of food on plate: 1  Total Meals Consumed: 8  Delivering Meal #9  Total amount of food on plate: 1  Total Meals Consumed: 9  Delivering Meal #10  Total amount of food on plate: 1  Total Meals Consumed: 10  Delivering Meal #11  Total amount of food on plate: 1  Delivering Meal #12  Total amount of food on plate: 2  Delivering Meal #13  Total amount of food on plate: 3 | Delivering Meal #1  Total amount of food on plate: 1  Total Meals Consumed: 1  Delivering Meal #2  Total amount of food on plate: 1  Total Meals Consumed: 2  Delivering Meal #3  Total amount of food on plate: 1  Total Meals Consumed: 3  Delivering Meal #4  Total amount of food on plate: 1  Total Meals Consumed: 4  Delivering Meal #5  Total amount of food on plate: 1  Total Meals Consumed: 5  Delivering Meal #6  Total amount of food on plate: 1  Total Meals Consumed: 6  Delivering Meal #7  Total amount of food on plate: 1  Total Meals Consumed: 7  Delivering Meal #8  Total amount of food on plate: 1  Total Meals Consumed: 8  Delivering Meal #9  Total amount of food on plate: 1  Total Meals Consumed: 9  Delivering Meal #10  Total amount of food on plate: 1  Total Meals Consumed: 10  Delivering Meal #11  Total amount of food on plate: 1  Delivering Meal #12  Total amount of food on plate: 2  Delivering Meal #13  Total amount of food on plate: 3 | Delivering Meal #1  Total amount of food on plate: 1  Total Meals Consumed: 1  Delivering Meal #2  Total amount of food on plate: 1  Total Meals Consumed: 2  Delivering Meal #3  Total amount of food on plate: 1  Total Meals Consumed: 3  Delivering Meal #4  Total amount of food on plate: 1  Total Meals Consumed: 4  Delivering Meal #5  Total amount of food on plate: 1  Total Meals Consumed: 5  Delivering Meal #6  Total amount of food on plate: 1  Delivering Meal #7  Total amount of food on plate: 2  Delivering Meal #8  Total amount of food on plate: 3  Delivering Meal #9  Total amount of food on plate: 4  Delivering Meal #10  Total amount of food on plate: 5 |

This solution also works when the buffer > # to consume, but the result provides nothing that the above three do not. As for quirks to my solution, for some reason my producer thread feels the need to fill the buffer after the limit is reached. This randomly occurred during testing and cause has eluded me.