1. The trade-off for adjusting the size of the bounded buffer is balancing memory consumption with program performance. When testing the program with a small, bounded buffer size, I observed frequent blocking of threads and potential delays in producing and consuming items. On the other hand, testing the program with a large, bounded buffer size resulted in reduced blocking of threads, improved throughput, and decreased synchronization overhead. The best bounded buffer size for the fastest program execution on my computer would depend on factors like the characteristics of the threads and the system. Generally, a size that can handle bursts of activity without significantly exceeding memory requirements would be ideal.
2. a. AMD Ryzen 7 1700 Eight-Core Processor

   b CPU(s): 2

   c Mem: 4868

3. I wouldn't say that there are any massive issues with the code that isn't working. I was having some issues with using the process ID, pagemon since every time I used it and clicked "enter" only a black space would occur and only after I typed the value, I wanted the prime number to reach would it work. Besides that, issue there aren't any other concerning ones.
4. How I tested my program for deadlock was that I created a "test_buffer.c" file that had a *producer that produced items and put them in the buffer and a consumer that consumed items from the buffer and a main method that created the producer and consumer threads and waits for the threads to finish. After I ran the command "gcc -o test_bufffer test_buffer.c buffer.c -pthread" and after "./test_buffer". If the program executes without any problems, then there is no deadlock, else it would become unresponsive.