Brandon Fowler

lab5pc1 output

```
Producer adding 1 to buffer
Producer Sleeps While Waiting For Consumer
Consume 0
Buffer Empty. Consumer Sleeps
Producer adding 2 to buffer
Consume 1
Consume 0
Buffer Empty. Consumer Sleeps
Producer adding 3 to buffer
Producer Sleeps While Waiting For Consumer
Consume 2
Consume 1
Consume 0
Buffer Empty. Consumer Sleeps
Producer adding 4 to buffer
Producer Sleeps While Waiting For Consumer
Consume 3
Consume 2
Consume 1
Consume 0
Buffer Empty. Consumer Sleeps
Producer adding 5 to buffer
Producer Sleeps While Waiting For Consumer
Consume 4
Consume 3
Consume 2
Consume 1
Consume 0
Buffer Empty. Consumer Sleeps
Producer adding 6 to buffer
Producer Sleeps While Waiting For Consumer
Consume 5
Consume 4
Consume 3
Consume 2
Consume 1
Consume 0
Buffer Empty. Consumer Sleeps
Producer adding 7 to buffer
Producer Sleeps While Waiting For Consumer
Consume 6
Consume 5
Consume 4
Consume 3
Consume 2
```

Consume 1
Consume 0
Buffer Empty. Consumer Sleeps
Producer adding 8 to buffer
Producer Sleeps While Waiting For Consumer
continues through 100.....

lab5pc1 questions

a) pthread_cond_wait tells the thread to stop working until it is signaled to start again.

b) condp and condc seem like they are basically on/off booleans. When wait is called on condp in the producer, it is basically setting condp to off, and telling the producer to wait until condp is switched back on. Likewise when wait is called on condc in the consumer, condc is set to off, and the thread must wait until condc is set back to on via a signal before it resumes work.

lab5pc2 question

When pthread_create(&pro, 0, producer, NULL) and pthread_create(&con, 0, consumer, NULL) are switched, then the consumer starts before the producer, instead of the producer starting first as it does when the lines are not switched. This causes the consumer to start, then wait immediately, instead of consuming right away. Switching the join order does not seem to make a difference, nor does the destroy order.

lab5pc3 output

Producer adding 1 to buffer
Producer adding 2 to buffer
Producer adding 3 to buffer
Producer adding 4 to buffer
Producer adding 5 to buffer
Producer adding 6 to buffer
Producer adding 7 to buffer
Producer adding 8 to buffer
Producer adding 9 to buffer
Producer adding 10 to buffer
Producer adding 11 to buffer
Producer adding 12 to buffer
Producer adding 13 to buffer
Producer adding 14 to buffer
Producer adding 15 to buffer
Consume 15
Consume 14
Consume 13
Consume 12
Consume 11
Consume 10
Consume 9
Consume 8

Consume 7
Consume 6
Consume 5
Consume 4
Consume 3
Consume 2
Consume 1
Producer adding 1 to buffer
Producer adding 2 to buffer
Producer adding 3 to buffer
Producer adding 4 to buffer
Producer adding 5 to buffer
Producer adding 6 to buffer
Producer adding 7 to buffer
Producer adding 8 to buffer
Producer adding 9 to buffer
Producer adding 10 to buffer
Producer adding 11 to buffer
Producer adding 12 to buffer
Producer adding 13 to buffer
Consume 13
Consume 12
Consume 11
Consume 10
Consume 9
Consume 8
Consume 7
Consume 6
Consume 5
Consume 4
Consume 3
Producer adding 3 to buffer
Producer adding 4 to buffer
Producer adding 5 to buffer
Producer adding 6 to buffer
Producer adding 7 to buffer
Producer adding 8 to buffer
Producer adding 9 to buffer
Producer adding 10 to buffer
Producer adding 11 to buffer
Producer adding 12 to buffer
Producer adding 13 to buffer
Consume 13
Consume 12
Consume 11
Consume 10
Consume 9
Consume 8
Consume 7

Consume 6
Consume 5
Consume 4
Producer adding 4 to buffer
Producer adding 5 to buffer
Producer adding 6 to buffer
Producer adding 7 to buffer
Producer adding 8 to buffer
Producer adding 9 to buffer
Producer adding 10 to buffer
Producer adding 11 to buffer
Producer adding 12 to buffer
Producer adding 13 to buffer
Producer adding 14 to buffer
Producer adding 15 to buffer
Producer adding 16 to buffer
Producer adding 17 to buffer
Producer adding 18 to buffer
Producer adding 19 to buffer
Producer adding 20 to buffer
Producer adding 21 to buffer
Producer adding 22 to buffer
Consume 22
Consume 21
Consume 20
Consume 19
Consume 18
Consume 17
Consume 16
Consume 15
Consume 14
Consume 13
Consume 12
Consume 11

lab5pc4 output

Producer adding 1 to buffer
Consume 1
Producer adding 1 to buffer
Producer adding 2 to buffer
Producer adding 3 to buffer
Producer adding 4 to buffer
Producer adding 5 to buffer
Producer adding 6 to buffer
Producer adding 7 to buffer
Producer adding 8 to buffer
Consume 8
Producer adding 8 to buffer

Producer adding 9 to buffer
Producer adding 10 to buffer
Producer adding 11 to buffer
Consume 11
Consume 10
Consume 9
Consume 8
Consume 7
Consume 6
Consume 5
Consume 4
Producer adding 4 to buffer
Producer adding 5 to buffer
Producer adding 6 to buffer
Producer adding 7 to buffer
Producer adding 8 to buffer
Producer adding 9 to buffer
Producer adding 10 to buffer
Producer adding 11 to buffer
Consume 11
Producer adding 11 to buffer
Producer adding 12 to buffer
Consume 12
Consume 11
Producer adding 11 to buffer
Producer adding 12 to buffer
Producer adding 13 to buffer
Producer adding 14 to buffer
Producer adding 15 to buffer
Producer adding 16 to buffer
Consume 16
Producer adding 16 to buffer
Consume 16
Producer adding 16 to buffer
Producer adding 17 to buffer
Consume 17
Producer adding 17 to buffer
Consume 17

lab5pc4 question

I ran this program with four producers and two consumers. There were no problems and the behavior was as expected. The producers were given more time by the scheduler as expected, since there were more producers. This caused the consumers to fall behind right from the start as shown above. By the time the buffer became full, the producers began sleeping a lot, making it obvious that the consumers were not keeping up. As a final note I also tested it with 30 producers and one consumer. The results were as expected; everything ran as it was supposed to run, and the consumer fell behind much faster.