Answers Exercise 3

Task 1

i)

The output with 5 producers and 0 consumers is:

```
Producer produced: 1534507321, buffer position: 1
Producer produced: 1171885191, buffer position: 0
Producer produced: 93906833, buffer position: 2
Producer produced: 1880913188, buffer position: 3
Producer produced: 1626918669, buffer position: 4
Producer produced: 813268848, buffer position: 0
Producer produced: 1111910226, buffer position: 1
Producer produced: 1174213401, buffer position: 2
Producer produced: 1898119435, buffer position: 3
Producer produced: 1445350030, buffer position: 4
Producer produced: 1080500579, buffer position: 0
Producer produced: 1205686382, buffer position: 1
Producer produced: 867974659, buffer position: 2
Producer produced: 1033013534, buffer position: 3
Producer produced: 481134014, buffer position: 4
Producer produced: 1920510410, buffer position: 0
Producer produced: 728059674, buffer position: 1
Producer produced: 1476770413, buffer position: 2
Producer produced: 2135844037, buffer position: 3
Producer produced: 199523957, buffer position: 4
```

The problem is that the buffer will be overwritten; the producers don't stop producing when the buffer is full!

The output with 0 producers and 5 consumers is:

```
Consumer consumed: 0, buffer position: 0
Consumer consumed: 0, buffer position: 1
Consumer consumed: 0, buffer position: 2
Consumer consumed: 0, buffer position: 3
Consumer consumed: 0, buffer position: 4
Consumer consumed: -1, buffer position: 0
Consumer consumed: -1, buffer position: 1
Consumer consumed: -1, buffer position: 2
Consumer consumed: -1, buffer position: 3
Consumer consumed: -1, buffer position: 4
Consumer consumed: -1, buffer position: 0
Consumer consumed: -1, buffer position: 1
Consumer consumed: -1, buffer position: 2
Consumer consumed: -1, buffer position: 3
Consumer consumed: -1, buffer position: 4
Consumer consumed: -1, buffer position: 0
Consumer consumed: -1, buffer position: 2
Consumer consumed: -1, buffer position: 1
Consumer consumed: -1, buffer position: 3
Consumer consumed: -1, buffer position: 4
Consumer consumed: -1, buffer position: 0
Consumer consumed: -1, buffer position: 1
Consumer consumed: -1, buffer position: 2
Consumer consumed: -1, buffer position: 3
Consumer consumed: -1, buffer position: 4
Consumer consumed: -1, buffer position: 0
Consumer consumed: -1, buffer position: 1
Consumer consumed: -1, buffer position: 2
```

The problem is that the consumers consumes even if the buffer is empty!

ii)

After editing the code, the output with 5 producers and 0 consumers is:

```
Producer produced: 1277546584, buffer position: 0
Producer produced: 804715142, buffer position: 1
Producer produced: 2113838101, buffer position: 2
Producer produced: 950719586, buffer position: 3
Producer produced: 568877856, buffer position: 4
```

The 5 producers stop producing when the buffer is full, but nothing will ever be consumed. . .

The output with 0 producers and 5 consumers is "empty" because nobody produces something. . .

Output with 2 producers and 2 consumers:

```
Producer produced: 1415501505, buffer position: 0
Producer produced: 1722149391, buffer position: 1
Consumer consumed: 1415501505, buffer position: 0
Consumer consumed: 1722149391, buffer position: 1
Producer produced: 1490085254, buffer position: 2
Consumer consumed: 1490085254, buffer position: 2
Producer produced: 1450319603, buffer position: 3
Producer produced: 186830753, buffer position: 4
Consumer consumed: 1450319603, buffer position: 3
Producer produced: 1983687688, buffer position: 0
Producer produced: 1800176350, buffer position: 1
Consumer consumed: 186830753, buffer position: 4
Consumer consumed: 1983687688, buffer position: 0
Consumer consumed: 1800176350, buffer position: 1
Producer produced: 1886612574, buffer position: 2
```

The producer consumer model is working as it should! Produces until max limit and consumes maximum to finish the buffer elements.

Task 3

The error lies in the interpretation of integers as booleans in C. The function try_lock returns 0 if the requested mutex could be locked and 1 if the mutex is already locked by anoter thread. For this code to work we want to enter the if statements at lines 8 and 28 if the mutex could be locked. But because the inteher 0 is interpreted as false the if statements are never entered. If the locking failed an even worse situation arrises where one thread tries to access resources currently used by the other thread and later tries to unlock the mutex owned by the other thread.

Task 4

Bounded Buffer Problem

The bounded buffer problem is about a buffer of size n capable of storing n entities of data. Further there are two processes. The first is called the producer, the second one the Consumer. Both try to operate on the buffer. The producer creates new data entities and tries to append them to the buffer. The consumer tries to remove data entities from the buffer.

Readers-Writers Problem

Dining Philosophers Problem

Task 5

A Deadlock is caused when two proceses hold a resource each and wait for the other to release it's resource. BOth threads wait for each other indefinitely. No progress is done. Starvation occurs if processes have different priorities. If high priority tasks are constantly executed therefore blockig resources of lower priority ones the lower priority processes are blocked indefinitely or at least for a problematic amount of time.