Problem 2 Pg. 255. Show that solution 3 to the Too Much Milk problem is safe – that it guarantees that at most one roommate buys milk.

A screenshot of a cell phone

Description automatically generated

When either thread executes either noteA or noteB will have a true value, or milk will be >0. In any of these cases the other roommate will buy milk. There is a chance that they will both buy milk. This makes the state safe regardless of the outcome. This can also be seen from the solution from the book.

***Case 3:*** *(noteB = 0, milk = 0). We know that thread B must not currently be executing any of the lines marked* ***B1-B5****. We also know that either noteA == 1 or milk > 0 will be true from this time forward (noteA OR milk is also a stable property). This means that B cannot buy milk in the future (either the test at B1 or B2 must fail), which contradicts our assumption that both A and B buy milk.*

Problem 3 Pg. 255. Precisely describe the set of possible outputs that could occur when the program shown in Figure 5.5 is run.

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

The program will create 3 queues and then try and add in 50 items into each thread, it will then try and remove 20. Whenever thread 0 is done executing the program returns. That means that thread 0 will have 10 elements left and the remaining threads could have any number that were inserted or removed.

Problem 10 Pg. 257. In the readers/writers lock example for the function RWLock::doneRead why do we use writeGo.Signal rather than writeGo.broadcast?

Because signal sends a signal that the one single thread is ready to be moved to the ready list whereas the broadcast sends a message that all the threads are ready to be moved into the ready list. This would cause problems if the broadcast function was called rather than the signal.