Question 4

a) The operations that can be performed on a semaphore are:

down – if the semaphore’s value is more than 0, then decrement the semaphore and continue. If the value becomes 0, put the process to sleep.

Up – this increments the value of the semaphore. If one or more processes were sleeping on that semaphore that couldn’t complete an earlier down operation, one is randomly chosen by the system to complete its down

b)

User-level approach: In user mode, the kernel is not aware of the existence of threads so it operates as it usually does. So it would pick a process, say X, and gives X control for its quantum. The thread scheduler in X decides which thread to run, say X1. Since there are no clock interrupts, this thread may continue running as long as it wants. If it uses up the processes’ entire quantum the kernel will select another process to run. Because of the nature of being user mode, it will not affect any other processes.

Eg. If there was another process, say B, the sequence A1, A2, A3, A1, A2, A3 is possible but not A1, B1, A2, B2, A3, B3

Kernel-level approach: The kernel picks a particular thread to run. It doesn’t have to take into account which thread belongs to which process. The thread is given a quantum and is forcibly suspended if it exceeds the quantum.

c)

To enter a critical region:

registerLock = lock

lock = 1

while registerLock is 1:

registerLock = lock

lock = 1

return

To leave a critical region:

lock = 0

return

d)

The main defect of both Peterson’s and the TSL solutions stems from them both requiring busy waiting. This approach wastes CPU time but can also have some undesired effects such as the priority inversion problem. This is where there are say 2 processes, a high priority process H and a low priority process L. The scheduling rules are that H is run whenever it is in ready state. At a certain moment, with L in its critical region, H becomes ready to run. H begins busy waiting but since L is never scheduled while H is running, L never gets the change to leave its critical region so H loops forever.