A4_soln.pdf

Q1) Could you predict the output of the program before execution? (Yes/No)

No, the output cannot be predicted. The output will be a random number such that NITER <= Output <= 2*NITER. This is due to not having semaphores to block the thread 'requesting resources'. The threads will therefore execute in an arbitrary order, giving an unexpected value. Looking at the a4_badoutput.txt file, the output passed 1/3 times and failed 2/3 times, giving unpredictable values when failing that are in the range of NITER <= Output <= 2*NITER.

Q3) Do you think it is possible to become deadlocked?

Well, by the notes presented from the lectures, a deadlock can arise if all four conditions (mutual exclusion, hold and wait, no pre-emption, circular wait) hold simultaneously. So, in our example, Mutual exclusion and no pre-emption exist because only one process runs at a time and they wait until completion but hold and wait and circular wait do not exist. By the definition, this is enough to conclude there is no deadlock possible in our program. Furthermore, by the theorem discussed in class, "If a resource allocation graph does not contain a cycle then no processes are deadlocked". As seen by the graphs I made below, the processes can execute in two different orders. Either, T1 will enter the count function (request resources) first which is followed by T2 when done executing or vice versa. In both situations, there is never a cycle. Therefore, by this theorem as well, there cannot be a deadlock.