ELE709 - Real-Time Computer Control Systems
Lab 4 - Resource Sharing and Coordination

Name: Austin Luu

---

1. **Exercise 4.1:**

   (a) Explain why the example program lab4.c didn't work correctly?

   The original code failed to run due to how the code was organized and variables were shared. In particular the lack of semaphore and/or mutexes resulted in multiple threads sharing and altering the same global variables concurrently. As a result the variables values being pulled at random times with no order/task scheduling or exclusion as to which thread can use the variable led to the failure.

   (b) A mutex was suggested as a way to make the example program work correctly. What is the purpose for using the mutex in this case? In particular, what is being protected by this mutex?

   Mutex are a viable solution for this problem as it creates ownership and exclusion as to the order of access for which threads use the global variable almost creating exclusive usage of a global variable. It is used to hold a specific variable or resource for a certain thread to make use of. To elaborate, once a resource or variable is locked by a thread, only that thread can use it and unlock or allow other threads to use it, thereafter. In the case of my code the critical section (character printing) is being locked and only released after a successful printing session for other threads to use afterwards.

2. **Exercise 4.2:** Explain the logic of your program for this exercise. In particular, what are the predicates (associated with the condition variable) for Thread A and Thread B?

   To print a sequence of "A0 B1 A2 B3 A4 B5 A6 B7 A8 B9", a mutex and a conditional variable were utilized for ownership/exclusiveness of global variables and signaling of completion. The mutex variable used is named "mut". conditional variable used is "can". For the execution of the program, initially a task locks and prints the first letter 'A' and number '0', then the conditional signal is set and sent causing the program to jump towards the conditional wait sequence. In this sequence the letter 'B' is printed along with the number which is within an array, which would be '1' and is called through using the variables string_index and temp_index. This is then repeated until the completion of both arrays. The predicate is the conditional statement used to compare the value of tmp_index and the size of the array holding the various numbers,