CS4023 **Week10 Tutorial Solutions**

The following questions are taken from the exercises at the end of Chapter 6 of SGG (ed. 8)

1. Servers can be designed to limit the number of open connections. For example, a server may wish to have only N socket connections at any point in time. As soon as N connections are made, the server will not accept another incoming connection until an existing connection is released. Explain how semaphores can be used by a server to limit the number of concurrent connections. (**Q6.17**)

   Answer A semaphore is initialized to the number of allowable open socket connections. When a connection is accepted, the `acquire()` method is called; when a connection is released, the `release()` method is called. If the system reaches the number of allowable socket connections, subsequent calls to `acquire()` will block until an existing connection is terminated and the release method is invoked.

2. Show that, if the `wait()` and `signal()` semaphore operations are not executed atomically – and thus implemented using some type of critical section mechanism – then mutual exclusion may be violated. (**Q6.18**)

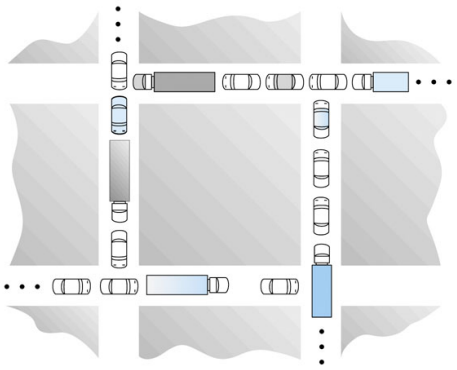
   Answer A `wait()` operation atomically decrements the value associated with a semaphore. If two `wait()` operations are executed on a semaphore when its value is 1, if the two operations are not performed atomically, then it is possible that both operations might proceed to decrement the semaphore value, thereby violating mutual exclusion.

3. Race conditions are possible in many computer systems. Consider a banking system with two functions: `deposit(amount)` and `withdraw(amount)`. These two functions are passed the amount that is to be deposited or withdrawn from a bank account. Assume a shared bank account exists between a husband and wife and concurrently the husband calls the `withdraw()` and the wife calls `deposit()`. Describe how a race condition is possible and what might be done to prevent the race condition form occurring. (**Q6.8**)

   Answer Assume the balance in the account is 250.00 and the husband calls `withdraw(50)` and the wife calls `deposit(100)`. Obviously the correct value should be 300.00. Since these two transactions will be serialized, the local value of balance for the husband becomes 200.00, but before he can commit the transaction, the `deposit(100)` operation takes place and updates the shared value of balance to 300.00. We then switch back to the

husband and the value of the shared balance is set to 200.00 - obviously an incorrect value.

4. For the traffic gridlock shown below, argue that the four necessary conditions for deadlock hold; give a simple rule for avoiding deadlocks in a system like this; and, most usefully, propose how Irish drivers can be encouraged to adhere to this rule. (**Q7.10**)



Answer

(a) The four necessary conditions for a deadlock are (1) mutual exclusion; (2) hold-and-wait; (3) no preemption; and (4) circular wait. The mutual exclusion condition holds since only one car can occupy a space in the roadway. Hold-and-wait occurs where a car holds onto its place in the roadway while it waits to advance in the roadway. A car cannot be removed (i.e. preempted) from its position in the roadway. Lastly, there is indeed a circular wait as each car is waiting for a subsequent car to advance. The circular wait condition is also easily observed from the graphic.

(b) A simple rule that would avoid this traffic deadlock is that a car may not advance into an intersection if it is clear it will not be able immediately to clear the intersection.