Solution to HW6

6.1:

Yes. As an example, consider \( c = 0 \) and \( x = 10 \), and let an update statement set the value of \( x \) to -5 after the condition \( x < c \) has been evaluated but before the statement \( y := x + 10 \) is executed. The result is \( y = 5 \). However, if we consider the update statement and the if statement as two operations, the effect of executing them concurrently should be either (i) \( y = 20 \) if the if statement is executed before the update statement, and (ii) \( y = 5 \) if the if statement is executed after the update statement.

6.2:

The upper bound is 100. The lower bound is not so obvious; it is only 2! The explanation is as follows: The statement \( \text{sum} := \text{sum} + 1; \) is implemented by a load-add-store sequence of instructions. Let one of the processes, say process \( P_1 \), perform a load and get delayed for a long time (presumably because its time slice elapses), and let process \( P_2 \) finish 49 iterations of its loop. The value of sum is now 49. Now, let process \( P_2 \) get delayed and let process \( P_1 \) finish its first iteration. The value of sum is now 1. Let \( P_2 \) start on its 50th iteration, execute the load instruction and get delayed (so it has a 1 in a CPU register). Now, let \( P_1 \) finish all iterations of its loop. The value of sum is now 50. Now let process \( P_2 \) finish its last iteration. The value of sum is now 2!

6.14:

Deadlocks do not arise if some philosopher is able to pick both left and right forks.

(a) If all philosophers pick up their left forks, the philosopher who has a vacant seat on her right will be able to pick up her right fork as well. When she finishes eating, the philosopher on her left will be able to pick up her right fork and so on, hence a deadlock will not arise. Analogously, one philosopher will be able to eat even when all philosophers first pick up their right forks.

(b) Let \( x \) be a left-handed philosopher whose right neighbor is a right-handed philosopher \( y \). If all philosophers try to pick up their first fork at the same time, then the fork between \( x \) and \( y \) will be free. So one of \( x \) and \( y \) will be able to pick up that fork and eat. When this philosopher finishes eating, the other philosopher can eat and so on, hence a deadlock will not arise.