Synchronization

1. [From [1], pg. 102, #3] Write a Hoare-style monitor that implements a bounded stack, $b$-stack, a stack of at most $\text{max}$ elements. The push and pop operations behave as follows:

   - $b$-stack.push($x$): if the stack is not completely full, the operation pushes the new element $x$ on top of the stack; otherwise, the operation has no effect and the value of $x$ is discarded.

   - $b$-stack.pop($x$): if the stack is not completely empty, the operation removes the value currently on top of the stack and returns it in the variable $x$; if the stack is empty, the operation waits until a value has been pushed on the stack and then completes by popping and returning this value.

2. [From [1], pg. 102, #4] Let a bounded semaphore $s$ be a general semaphore that cannot exceed a given value $m > 0$. The corresponding operations PB and VB are defined as:

   - PB($s$): wait until $s > 0$; then decrement $s$ by 1.
   - VB($s$): wait until $s < m$; then increment $s$ by 1.

   Write a Hoare-style monitor, named $s$, such that the calls $s$.PB() and $s$.VB() emulate the operations on bounded semaphores.

3. [From [1], pg. 103, #12]

   The following solution to the readers/writers problem was proposed by Courtois, Heymans, and Parnas (1971):

   ```c
   reader() {
     P(mutex);
     read-cnt++;
     if (read-cnt==1) P(w);
     V(mutex);
     READ;
     P(mutex);
     read-cnt--;
     if (read-cnt==0) V(w);
     V(mutex);
   }
   writer() {
     P(w);
     WRITE;
     V(w);
   }
   ```

   Initially: mutex=1; read-cnt=0; w=1;

   Does this solution satisfy the basic requirements of the readers/writers problem? Is starvation of readers or writers possible?

Implementing Processes and Threads
1. [From [1], pg. 145, #9] Consider a machine instruction, TSB, which performs the following function as a single indivisible operation:

\[ \text{TSB}(X, L) \text{ if } (X==0) \text{ goto } L \text{ else } X = 0; \]

I.e., TSB tests the variable X. Depending on the outcome of the test, the instruction either branches to an address L, or it sets X to zero, and execution continues with the next instruction following TSB. Implement the Pb and Vb operations on binary semaphores using TSB.

REFERENCES