Multithreaded Programming with ThreadMentor
Self Assessment Pretest

Name: _______________________

**General Directions:** We would like you to assess your current level of understanding regarding multithreaded programming concepts and skills and your understanding of handling threads and synchronization primitives. This information will be helpful to us as we determine the appropriate level for teaching these materials and preparing for future topics, but will not influence your course grade.

What is your understanding of the following areas, topics, or concepts:

1. The skills of creating, terminating, and joining threads
   - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent
2. The impact of using threads on program efficiency
   - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent
3. The differences between kernel-supported and user-level threads
   - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent
4. The differences between non-preemptive (cooperative) scheduling and preemptive thread scheduling
   - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent
5. The concept and correct use of mutual exclusion locks
   - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent
6. The concept and correct use of semaphores
   - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent
7. The concept and correct use of critical sections
   - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent
8. The construction of critical sections with locks and/or semaphores
   - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent
9. The concept and correct use of monitors
   - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent
10. The four states of a thread that are involved in a monitor call
    - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent
11. The two commonly used monitor styles
    - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent
12. The differences between the monitor discussed in your text and Java monitors
    - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent
13. The concept and correct use of barriers
    - [ ] none  [ ] low  [ ] good  [ ] very good  [ ] excellent

— over please —
14  The concept and correct use of readers-writers locks
   □ none  □ low  □ good  □ very good  □ excellent
15  The differences between reader-priority and writer-priority
    readers-writers locks
   □ none  □ low  □ good  □ very good  □ excellent
16  The concept and correct use of synchronous channels
   □ none  □ low  □ good  □ very good  □ excellent
17  The concepts and correct use of asynchronous channels
   □ none  □ low  □ good  □ very good  □ excellent
18  The differences between synchronous and asynchronous channels
   □ none  □ low  □ good  □ very good  □ excellent
19  The concept and the impact on program correctness of race conditions
   □ none  □ low  □ good  □ very good  □ excellent
20  How confident are you in pinpointing a potential race condition?
   □ none  □ low  □ good  □ very good  □ excellent
21  The distinction between busy waiting and waiting using locks,
    semaphores and/or condition variables
   □ none  □ low  □ good  □ very good  □ excellent
22  The concept of deadlock
   □ none  □ low  □ good  □ very good  □ excellent
23  How confident are you in pinpointing a potential deadlock?
   □ none  □ low  □ good  □ very good  □ excellent
24  The differences between thread models and process models
   □ none  □ low  □ good  □ very good  □ excellent

Please fill in your answers for the following questions:

1. Have you used Java and its thread capability?

2. Have you used threads under any operating system (e.g., Windows, Linux, Unix, etc.)? Please specify.
3. Do you have any desire to do more than two tasks at the same time on your computer (e.g., word processing and programming)?
   Please elaborate.

4. Do you have any desire to do more than two tasks at the same time in a single program?
   Please elaborate.

5. Why are you learning multithreaded programming?

6. What is your expectation of learning multithreaded programming?
   Please elaborate.