

# Threads & Concurrency



## Practice Exercises

- 4.1 Provide three programming examples in which multithreading provides better performance than a single-threaded solution.
- 4.2 Using Amdahl's Law, calculate the speedup gain of an application that has a 60 percent parallel component for (a) two processing cores and (b) four processing cores.
- 4.3 Does the multithreaded web server described in Section 4.1 exhibit task or data parallelism?
- 4.4 What are two differences between user-level threads and kernel-level threads? Under what circumstances is one type better than the other?
- 4.5 Describe the actions taken by a kernel to context-switch between kernel-level threads.
- 4.6 What resources are used when a thread is created? How do they differ from those used when a process is created?
- 4.7 Assume that an operating system maps user-level threads to the kernel using the many-to-many model and that the mapping is done through LWPs. Furthermore, the system allows developers to create real-time threads for use in real-time systems. Is it necessary to bind a real-time thread to an LWP? Explain.

