



## Chapter 4 Important Points

1. Differences between representation of a **thread** and a **process**; creation and implementation differences as well
2. Benefits of **multi-threading** over **single threading** in each of the following categories; examples of each:
  - (a) Responsiveness
  - (b) Resource Sharing
  - (c) Economy
  - (d) Scalability
3. Components of a thread (what resources it uses)
4. Challenges of programming multi-threaded applications
  - Dividing activities
  - Balance
  - Data splitting
  - Data dependency
  - Testing and debugging
5. Difference between **parallelism** and **concurrency**
6. **Amdahl's Law** and its applications
7. Difference between **data parallelism** and **task parallelism** and examples of each
8. Threading models
  - **User threads** versus **kernel threads** - what are they, how are they different
  - **Many to one** thread model
  - **One to one** thread model
  - **Many to many** thread model
  - **Two-level thread** model
9. Thread libraries - what are they and what are examples
  - **Asynchronous** versus **synchronous** threading
  - **POSIX threads** - what it is
10. Implicit threading -
  - (a) definition, examples
  - (b) thread pools
  - (c) **OpenMP** - what is it and how is it used
11. Threading issues
  - (a) Interaction between **fork()** and multi-threading
  - (b) How **exec()** works with multiple threads in a process
  - (c) What are **signals** and what is the problem with sending and handling signals with multi-threaded programs
  - (d) **Thread cancellation** - types of cancellation and how it is handled