**Separation of policy from mechanism/** **Optimizing for the Most Common or Important Case**

Separation of policy from mechanism

* One important principle is the separation of policy from mechanism.
  + Mechanisms determine **how** to do something; policies determine **what** will be done.
  + For example, the timer construct is a mechanism for ensuring CPU protection, but deciding how long the timer is to be set for a particular user is a policy decision.
* The separation of policy and mechanism is important for flexibility. Policies are likely to change across places or over time.
* For instance, consider a mechanism for giving priority to certain types of programs over others. If the mechanism is properly separated from policy, it can be used to support a policy decision that I/O-intensive programs should have priority over CPU-intensive ones or to support the opposite policy.
* Policy decisions are important for all resource allocation. Whenever it is necessary to decide whether or not to allocate a resource, a policy decision must be made. Whenever the question is **how** rather than **what**, it is a mechanism that must be determined.

## Optimizing for the Most Common or Important Case

Huffman coding is an example of this principle. It is a form of compression that encodes each symbol with a variable-length quantity, where the shorter quantities are used to encode more frequently occurring symbols.