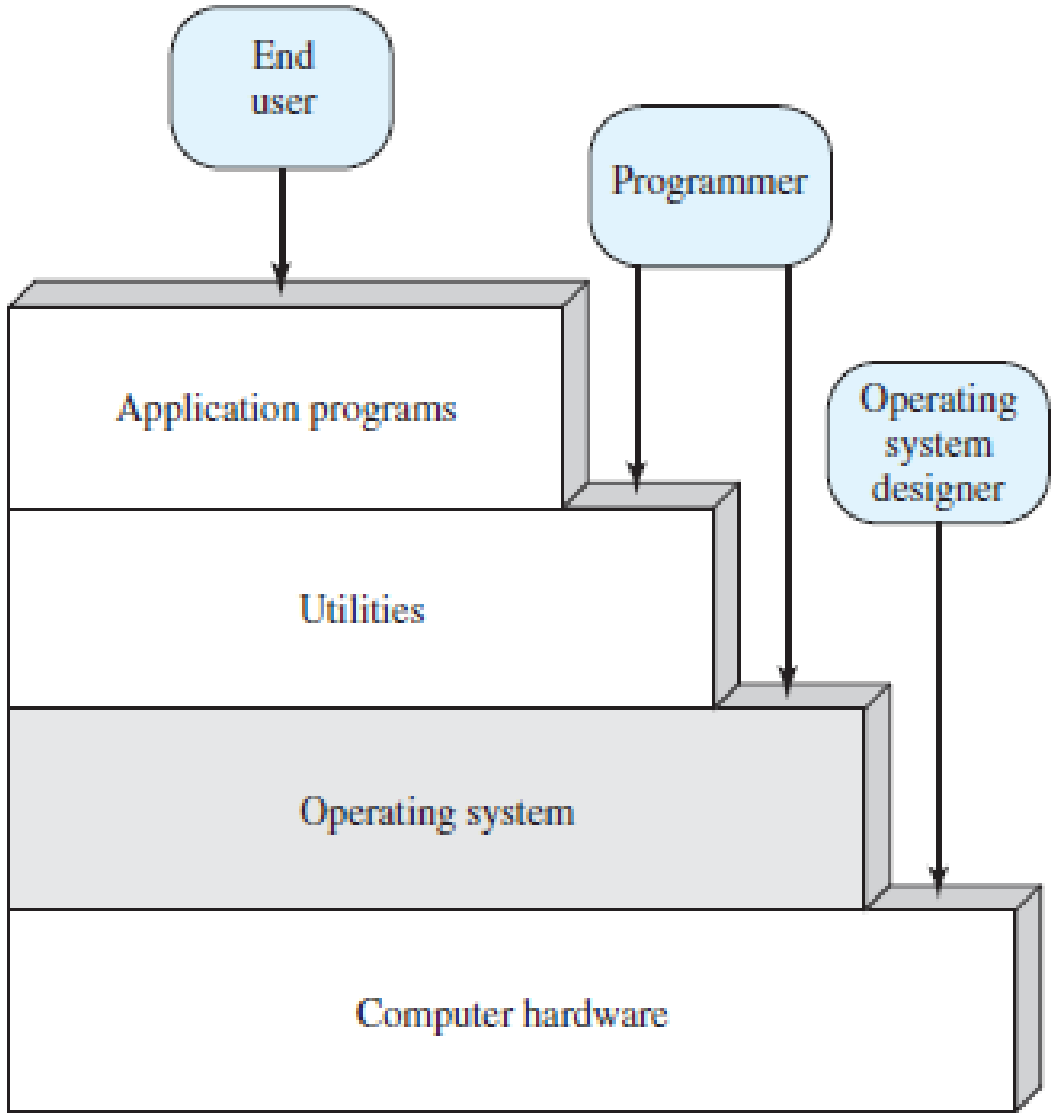
سیستم عامل دوره کاردانی



Layers and Views of a Computer System

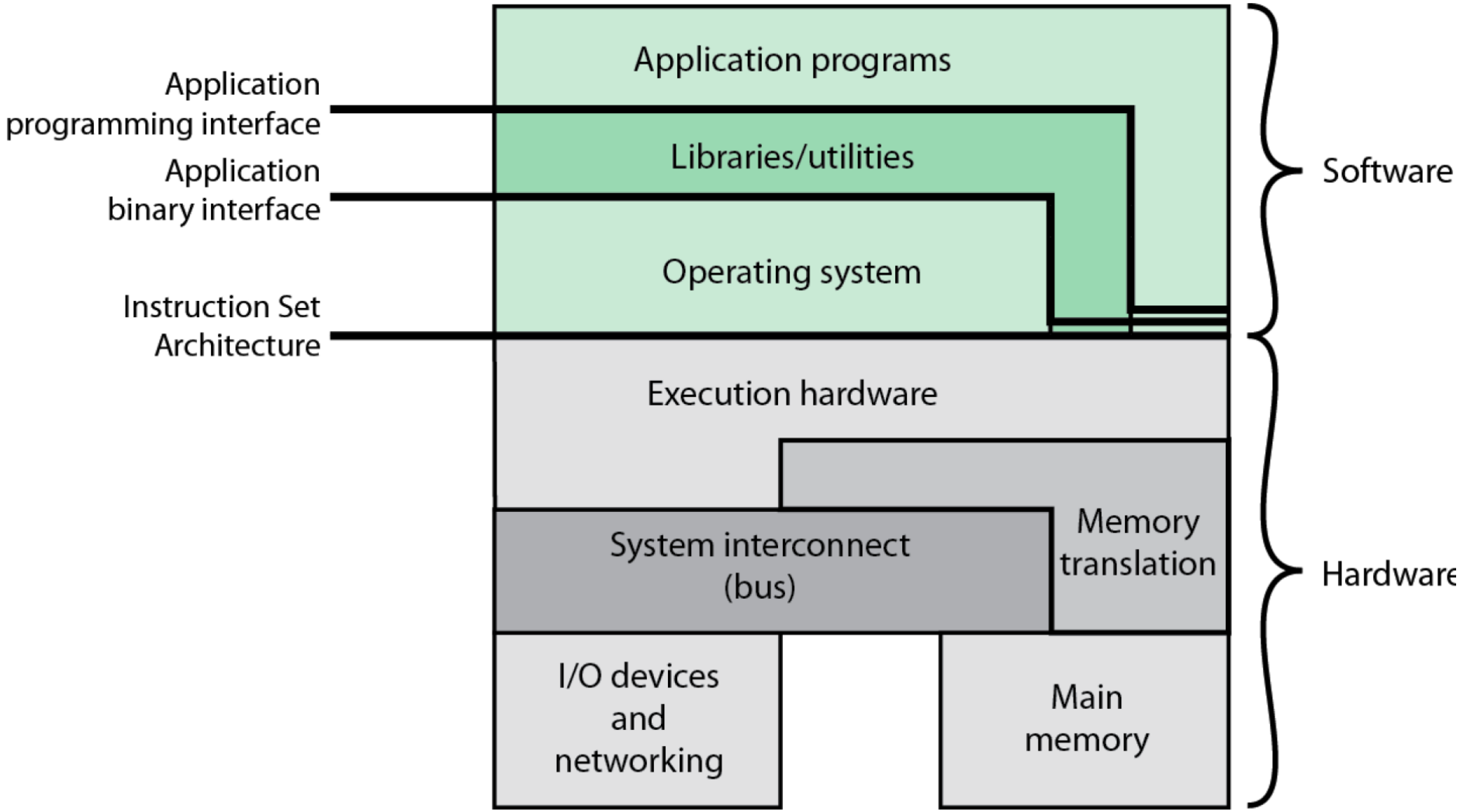


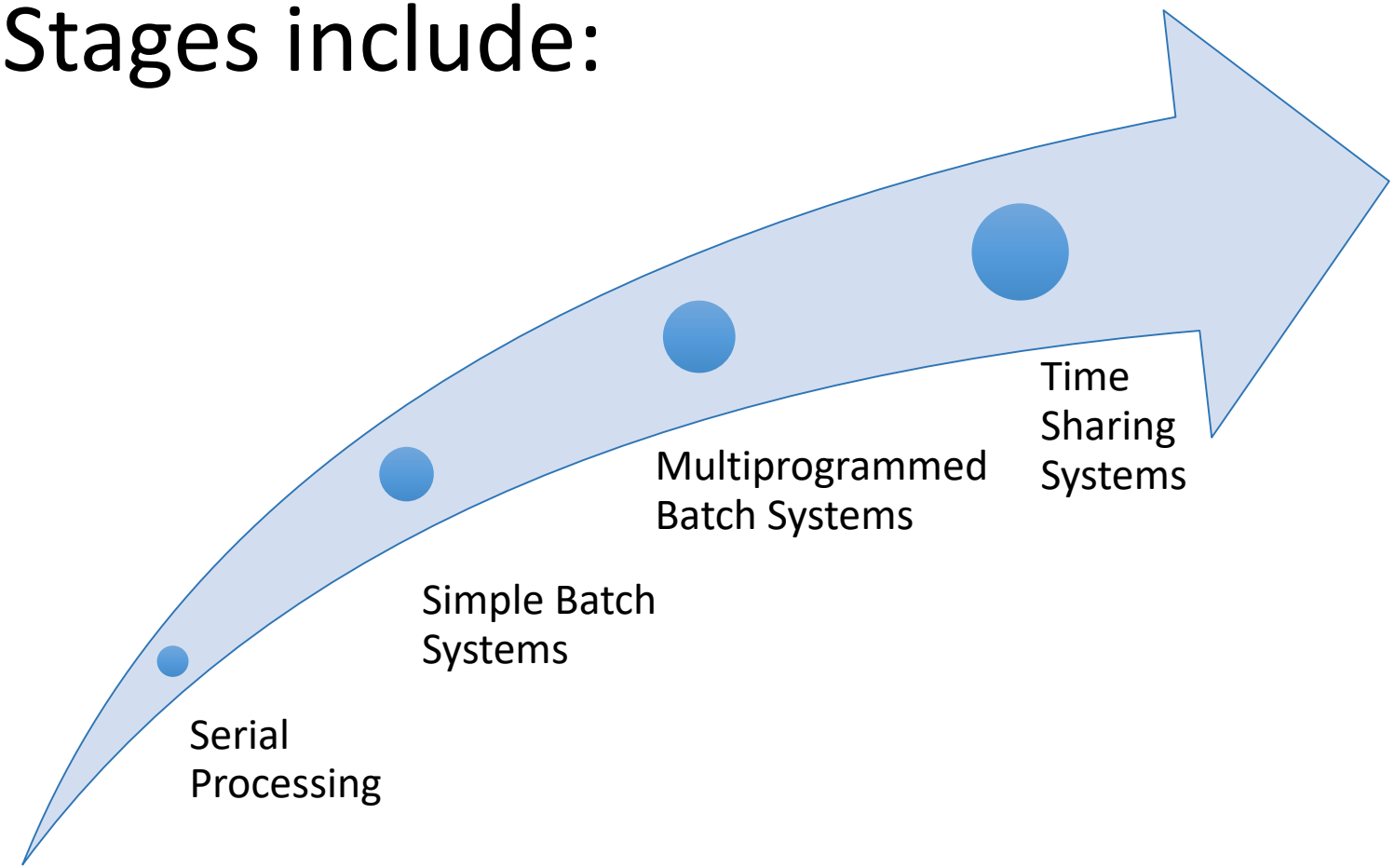
Figure 2.1 Computer Hardware and Software Infrastructure

Operating System Services

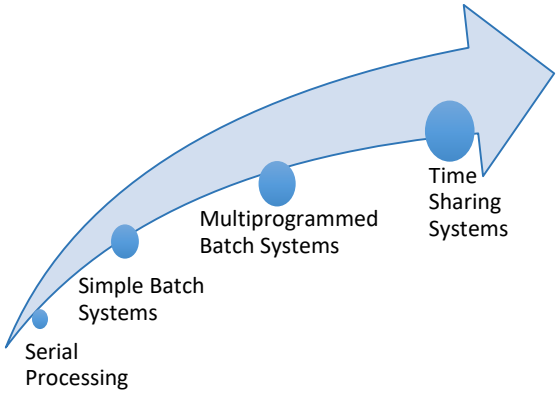
- Program development
- Program execution
- Access I/O devices
- Controlled access to files
- System access
- Error detection and response
- Accounting

Evolution of Operating Systems

- Stages include:



Simple Batch Systems



- Early computers were very expensive
  - important to maximize processor utilization
- Monitor
  - user no longer has direct access to processor
  - job is submitted to computer operator who batches them together and places them on an input device
  - program branches back to the monitor when finished

Monitor Point of View

- Monitor controls the sequence of events
- *Resident Monitor* is software always in memory
- Monitor reads in job and gives control
- Job returns control to monitor

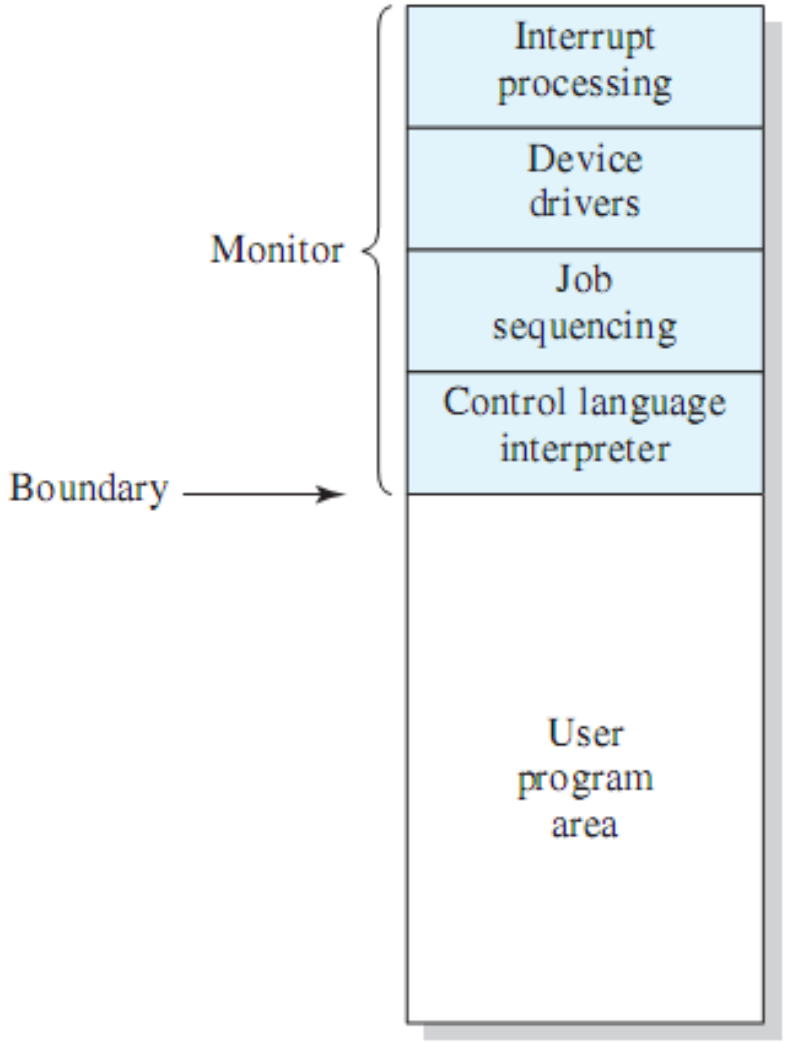
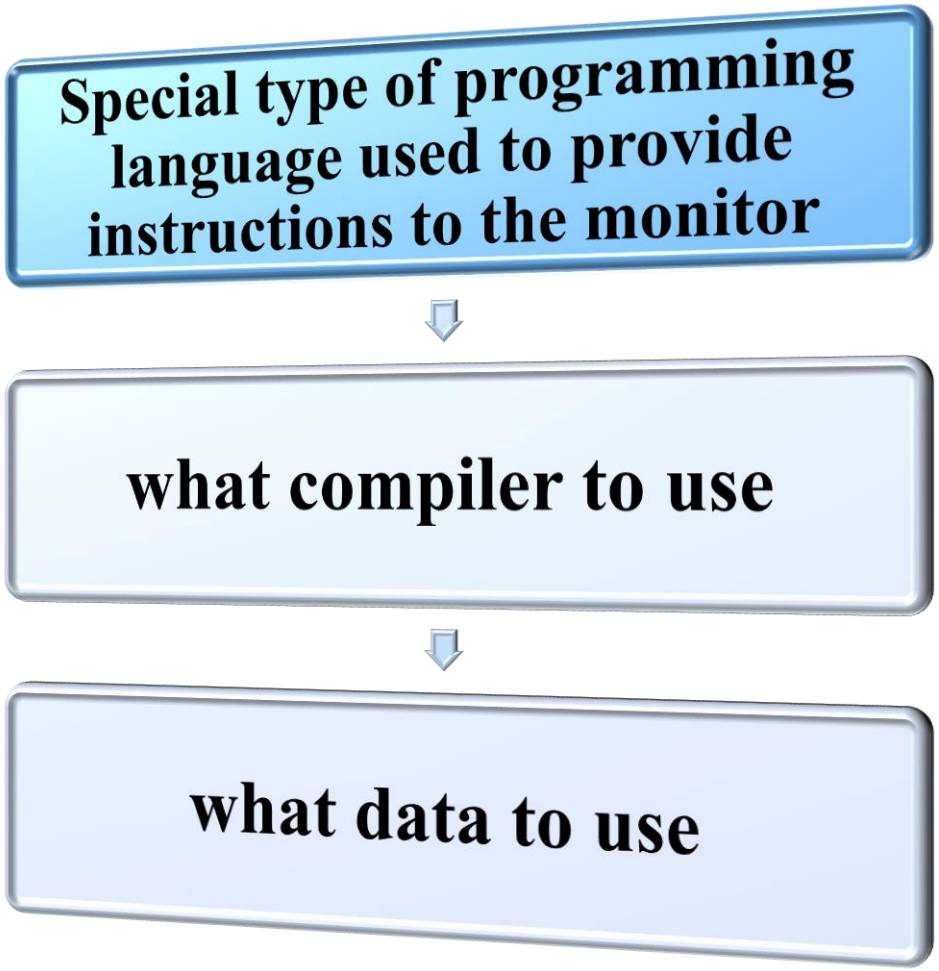


Figure 2.3 **Memory Layout for a Resident Monitor**

Processor Point of View

- Processor executes instruction from the memory containing the monitor
- Executes the instructions in the user program until it encounters an ending or error condition
- "*control is passed to a job*" means processor is fetching and executing instructions in a user program
- "*control is returned to the monitor*" means that the processor is fetching and executing instructions from the monitor program

Job Control Language (JCL)



Modes of Operation

User Mode

- user program executes in user mode
- certain areas of memory are protected from user access
- certain instructions may not be executed

Kernel Mode

- monitor executes in kernel mode
- privileged instructions may be executed
- protected areas of memory may be accessed

Multiprogrammed Batch Systems

Read one record from file	15 μs
Execute 100 instructions	1 μs
Write one record to file	15 μs
TOTAL	31 μs

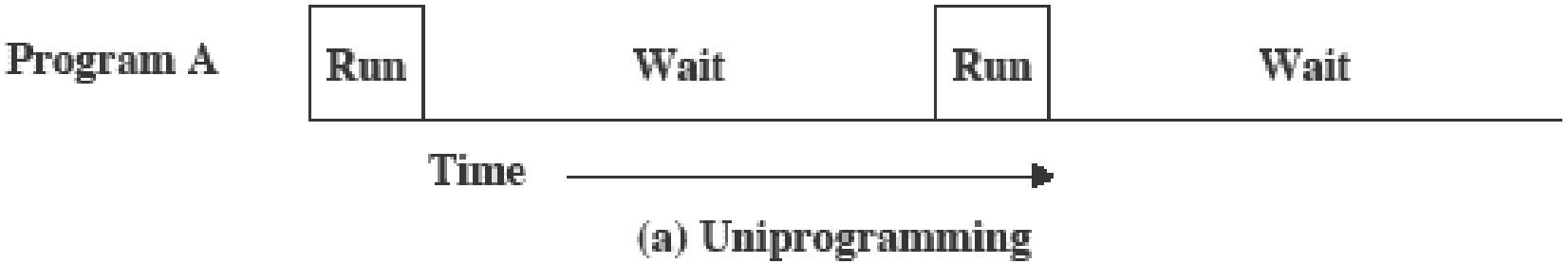
Percent CPU Utilization $= \frac{1}{31} = 0.032 = 3.2\%$

Figure 2.4 System Utilization Example

Processor is often idle

- even with automatic job sequencing
- I/O devices are slow compared to processor

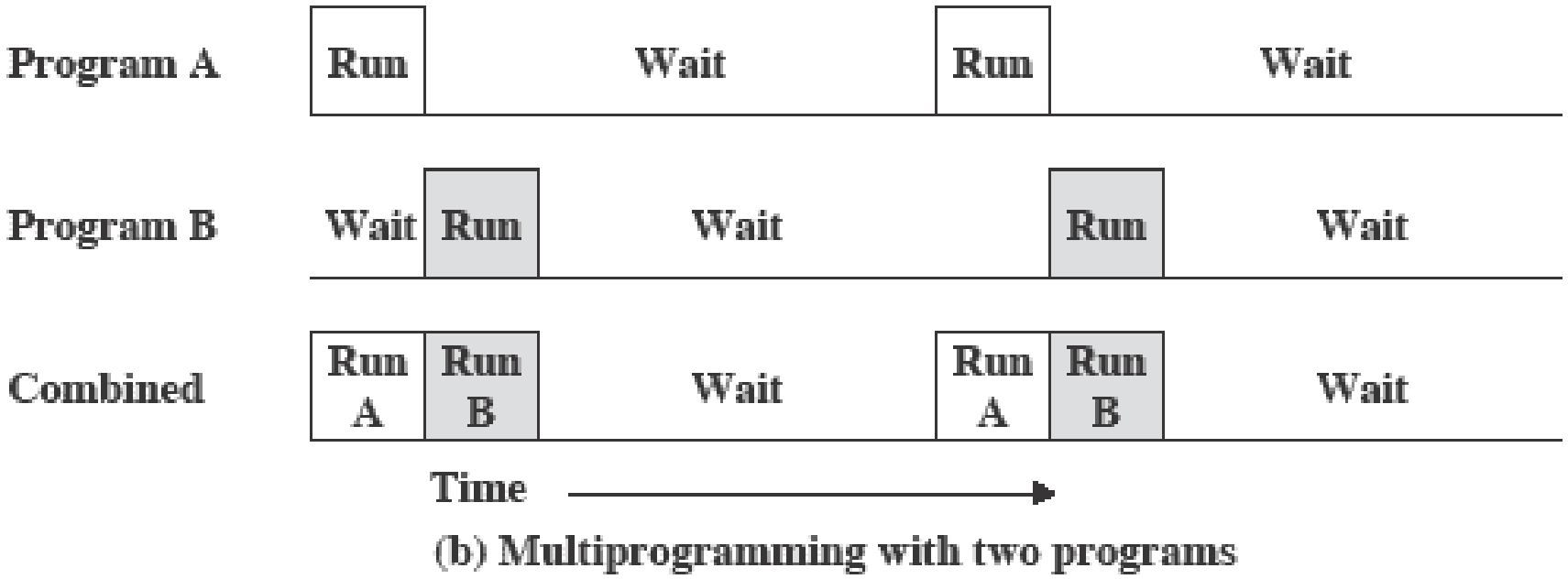
Uniprogramming



(a) Uniprogramming

- The processor spends a certain amount of time executing, until it reaches an I/O instruction; it must then wait until that I/O instruction concludes before proceeding

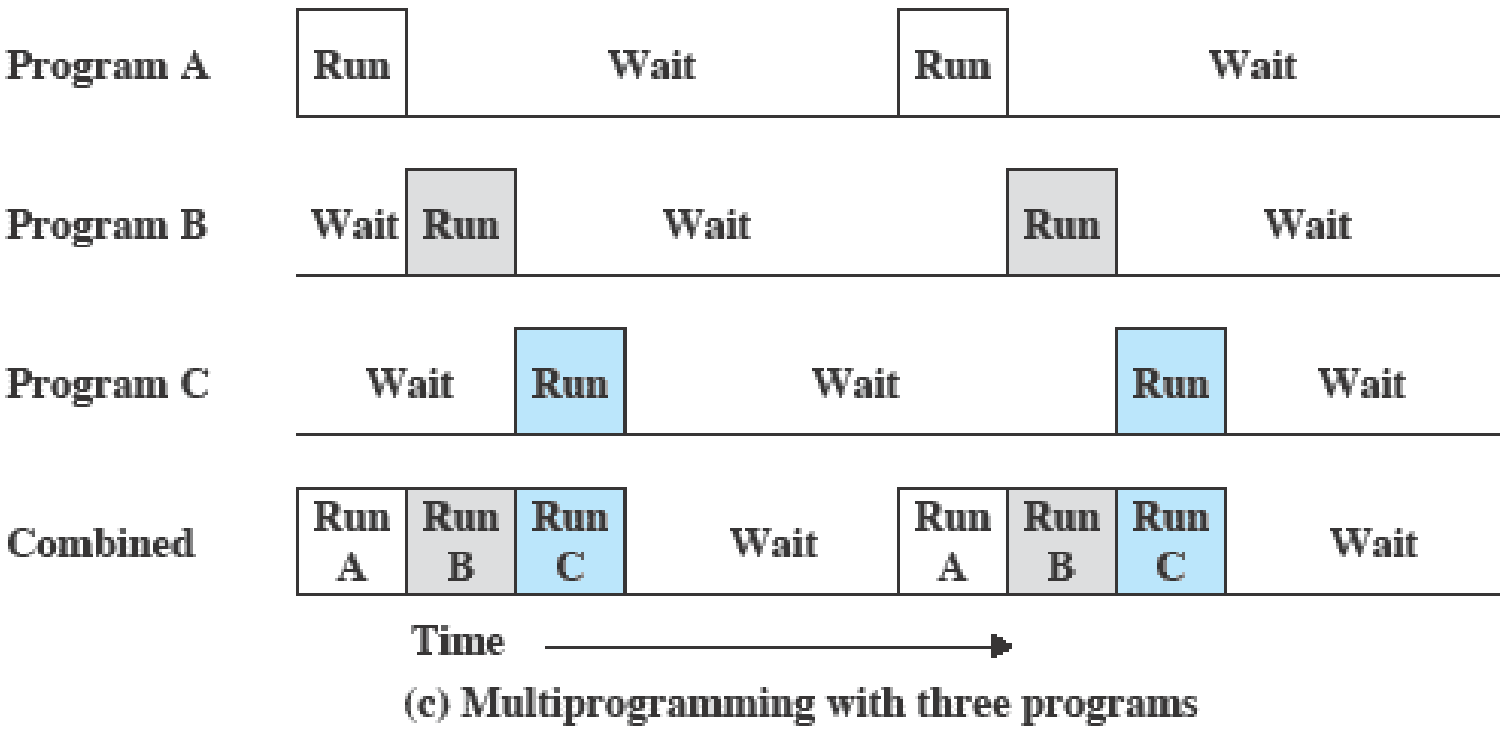
Multiprogramming



(b) Multiprogramming with two programs

- **There must be enough memory to hold the OS (resident monitor) and one user program**
- **When one job needs to wait for I/O, the processor can switch to the other job, which is likely not waiting for I/O**

Multiprogramming



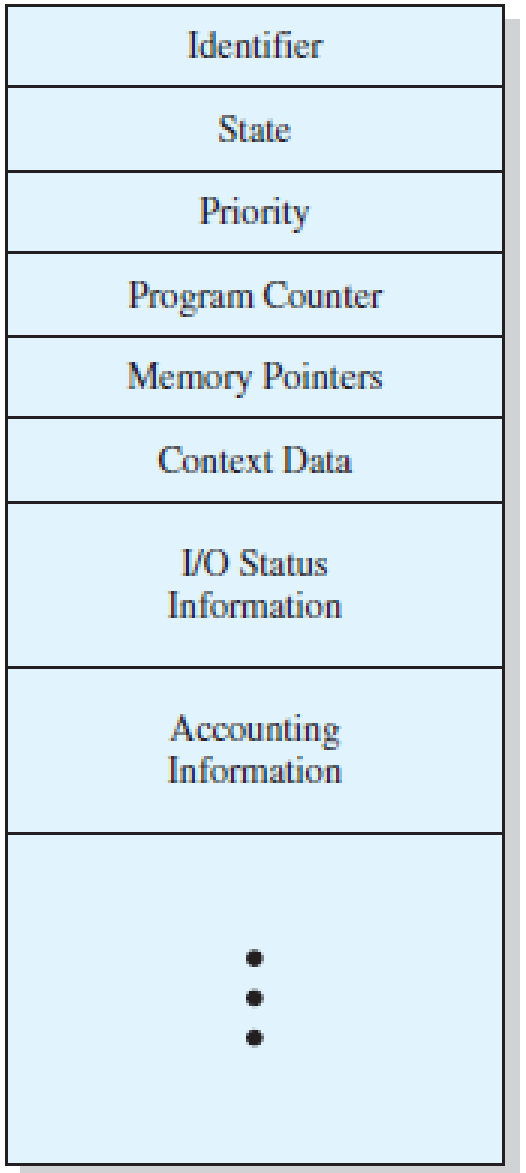
(c) Multiprogramming with three programs

Multiprogramming

- also known as multitasking
- memory is expanded to hold three, four, or more programs and switch among all of them

number of elements, including the following:

- **Identifier:** A unique identifier associated with this process, to distinguish it from all other processes.
- **State:** If the process is currently executing, it is in the running state.
- **Priority:** Priority level relative to other processes.
- **Program counter:** The address of the next instruction in the program to be executed.
- **Memory pointers:** Includes pointers to the program code and data associated with this process, plus any memory blocks shared with other processes.
- **Context data:** These are data that are present in registers in the processor while the process is executing.
- **I/O status information:** Includes outstanding I/O requests, I/O devices (e.g., tape drives) assigned to this process, a list of files in use by the process, and so on.
- **Accounting information:** May include the amount of processor time and clock time used, time limits, account numbers, and so on.



Simplified Process Control Block