1. **Quản lý bộ nhớ**

**Mục đích:** phân phối, bảo vệ và tối ưu sử dụng bộ nhớ chính giữa các tiến trình, cung cấp không gian ảo để vượt quá giới hạn RAM vật lý.

**Cách hoạt động**: Phân phối RAM cho tiến trình, dùng bộ nhớ ảo (virtual memory) để vượt giới hạn RAM, bảo vệ vùng nhớ giữa các tiến trình.

**Ví dụ thực tế:** Windows dùng pagefile.sys, Linux dùng swap để lưu bớt dữ liệu khi RAM đầy.

**Tóm tắt ngắn:** Quản lý bộ nhớ đảm bảo mỗi tiến trình có không gian riêng, cho phép chạy nhiều tiến trình vượt quá RAM vật lý nhờ bộ nhớ ảo, đồng thời bảo vệ và tối ưu hiệu năng bằng cơ chế bảng trang và thuật toán thay thế trang.

1. **Quản lý thiết bị nhập/xuất**

**Mục đích:** ẩn chi tiết phần cứng, điều phối truy cập tới thiết bị (bàn phím, ổ đĩa, mạng, màn hình, ...), xử lý ngắt và cung cấp driver/điều khiển.

**Cách hoạt động:** Sử dụng driver để giao tiếp với phần cứng, xử lý yêu cầu qua ngắt (interrupt) và DMA để giảm tải CPU.

**Ví dụ thực tế:** Khi in tài liệu, dữ liệu được đưa vào hàng đợi (spooler) rồi chuyển tới máy in qua driver.

**Tóm tắt ngắn:** Quản lý I/O ẩn chi tiết phần cứng bằng driver và controller, dùng ngắt và DMA để tối ưu CPU, quản lý hàng đợi yêu cầu và đảm bảo đồng bộ/khôi phục lỗi khi truy cập thiết bị.

**3. Cung cấp giao diện người dùng**

**Mục đích:** cho phép người dùng tương tác với hệ thống - nhập lệnh, quản lý file, khởi chạy chương trình, giám sát trạng thái hệ thống.

**Cách hoạt động:** Cung cấp CLI (gõ lệnh qua shell) và GUI (cửa sổ, biểu tượng, menu) để người dùng thao tác.

**Ví dụ thực tế:** Windows Explorer (GUI) để duyệt file, PowerShell (CLI) để chạy lệnh quản trị.

**Tóm tắt ngắn:** Giao diện người dùng - CLI và GUI là lớp tương tác giữa con người và OS; CLI mạnh về tự động hoá và quản trị, GUI thân thiện cho người dùng phổ thông và được xây dựng bằng bộ phận đồ họa, trình quản lý cửa sổ và API hệ thống.

**4. Quản lý hệ thống tệp**

**Mục đích:** cung cấp cấu trúc lưu trữ, metadata, bảo toàn dữ liệu, và giao diện truy cập.

**Cách hoạt động:** Tổ chức file/thư mục, quản lý quyền truy cập, bảo vệ dữ liệu bằng journaling hoặc copy-on-write.

**Ví dụ thực tế:** NTFS trên Windows, ext4 trên Linux đảm bảo dữ liệu không mất khi mất điện.

**Tóm tắt ngắn:** Quản lý hệ thống tệp tổ chức dữ liệu trên đĩa, đảm bảo quyền truy cập, hiệu năng thông qua cache, và tính toàn vẹn dữ liệu bằng journaling hoặc cơ chế tương tự.

1. **Quản lý tiến trình**

**Mục đích:** tạo, lên lịch, giám sát và kết thúc tiến trình/tiến trình con; đảm bảo CPU được chia sẻ công bằng và hiệu quả; xử lý đồng bộ & giao tiếp giữa tiến trình.

**Cách hoạt động:** Tạo, huỷ, và lên lịch tiến trình; chuyển đổi ngữ cảnh để nhiều tiến trình cùng chạy; hỗ trợ đồng bộ & giao tiếp.

**Ví dụ thực tế:** Linux dùng CFS scheduler để chia CPU công bằng giữa các chương trình đang chạy.

**Tóm tắt ngắn:** Quản lý tiến trình điều phối tạo/hủy tiến trình, phân chia CPU theo thuật toán scheduling, thực hiện context switching và hỗ trợ cơ chế đồng bộ/IPC để tiến trình phối hợp mà không xung đột.