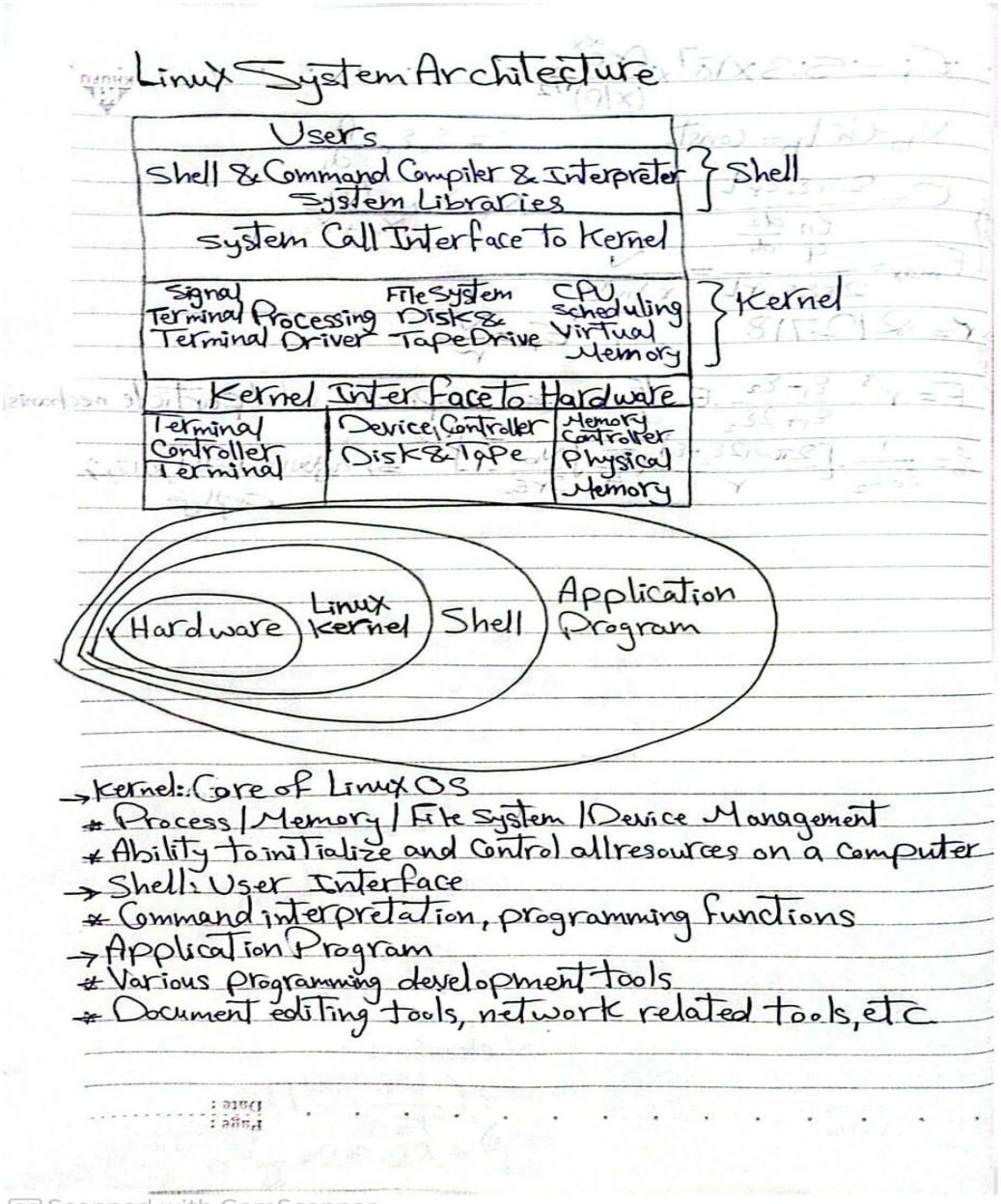
1. **Draw or describe the Linux architecture layers (hardware → kernel → shell → user space). Where do system calls fit?**



- System calls are the *bridge between user space and the kernel*, allowing user programs to request services like file I/O, networking, and process creation.

2. Explain the purpose of these directories: /, /bin, /sbin, /usr, /etc, /var.

- / (Root directory)
    i. The top of the filesystem hierarchy.
- /bin (Essential user binaries)
    i. Binary executables needed for basic system operation.
    ii. Contains programs required for all users, even in single-user mode or maintenance.
- /sbin (System binaries)
    i. Like /bin, but contains system administration tools (mostly for root).
    ii. Used for system repair, configuration, and management.
- /usr (User programs & data)
    i. Contains the bulk of user-space software.
- /etc (Configuration files)
    i. System-wide configuration files (text files).
    ii. No binaries here.
- /var (Variable data)
    i. Contains files that change frequently during system operation.

3. Why does Linux treat everything as a file? Explain the difference between a program and a process.

A. Uniform interface: Programs don't need to care whether they're reading from a text file, a hard disk, or a keyboard → they just use the same system calls: open(), read(), write(), close().
B. Simplicity: One consistent API makes the OS easier to design and applications easier to write.
C. Flexibility: You can redirect output of one program to another because everything is file-like.

- **Program**
    i. A file stored on disk (usually in /bin, /usr/bin, etc.).
    ii. Contains instructions + static data.
- **Process**
    i. A program in execution.
    ii. When you run a program, the OS:
        1. Loads it into memory
        2. Creates a process control block (PCB)
        3. Assigns it a PID (Process ID)
        4. Manages its execution (scheduling, memory, I/O)