Note on Chapter 1 and 2

**Chapter 1:**

* **Three key ideas** of Operating System: Visualization, Concurrency and Persistence.
* **What we will be learning**: how an operating system works, what program to run next on CPU, how it handles memory overload in a virtual memory system, how virtual machine monitors work.
* **Favorite quote**: ’I hear and I forget. I see and I remember. I do and I understand.’

**Chapter 2:**

* **Von Neumann model of computing**: When a program runs, it executes instructions. Many millions or billions of times every second, the processor fetches an instruction from memory, decodes it (figures out which instruction) and executes it (does the thing that the instruction tells it to do). Then, it moves on to the next instruction.
* **The body of software** is responsible for making it easy to run programs, allowing them to share memory, enabling programs to interact with devices and other stuff. **The body of software is called operating system** as it is in charge of making sure the system operates correctly and efficiently in an easy-to-use manner. The way the OS does this is through a technique called **virtualization**. The OS takes a physical resource and transforms it into a more general, powerful and easy-to-use virtual form of itself. We sometimes refer to the OS as a **virtual machine**.
* The OS provides some interfaces that we can call. It exports a few hundred system calls that are available to applications. We sometimes say that the OS provides a **standard library** to applications because it provides these calls to run programs, access memory and devices, and other actions.
* Because virtualization allows many programs to run, and many programs to concurrently access their own instructions and data, and many programs to access devices, the OS is sometimes known as a **resource manager**. The **resource** here refers to the CPU, memory and disk, and the OS’s role is to **manage** those resource.
* Many programs can be run on a single processor system (**CPU**). This is because OS creates an **illusion** that the system has a very large number of virtual CPUs. We called it **virtualizing the CPU**.
* The model of physical memory in modern memory is simple. Memory is just an array of bytes. To read memory, one must specify the address to access the data stored there. To write memory, we must specify the data to be written to given address.
* Running programs have **unique process identifier** (**PID**).
* A **hard drive** is a common repository for long-lived information. **Solid-state drives (SSDs)** are also be used as well.
* The software in the OS that usually manages the disk is called the **file system.**
* In summary, what OS does is that it takes **physical resources**, such as a CPU, memory, or disk, and **virtualizes** them. It handles tough and tricky issues related to **concurrency**. And it stores files **persistently**, thus making them safe over the long-term.
* Designing goals:

1. Provide **high performance**, or to **minimize the overheads**, which can be extra time and extra space.
2. Provide **protection** between applications, as well as between the OS and applications: make sure that we can run many programs at the same time, but they don’t harm others and the OS itself => principle of **Isolation**.
3. OS must also **run non-stop** or high degree of **reliability**.
4. Other goals can be **energy-efficiency, security, mobility**, etc.

* Other aspects that are not discovered: **networking, graphics** and **security.**