✅ 四个环境的区别

环境名称	类型	用途	操作系统	是否必须
moss	EAIT Linux 学生服务器	编写、提交大部分课程代码（Linux相关）	Linux	✅ **必用**
lichen	EAIT OpenBSD 学生服务器	用于完成 OpenBSD 系统相关实验和练习（如内核编译）	OpenBSD	🟡 **可选 / 推荐使用**
COMP3301 VM Control	控制你创建的 VM 的 Web 页面	用来远程创建 / 删除 / 控制你的虚拟机	-	✅
COMP3301 VM	你的个人虚拟机（可选为 Linux / BSD）	你亲自操作的操作系统实验环境	自选（Linux/OpenBSD）	✅

重点脉络速览

模块	关键概念	一句话记忆
操作系统定位	“资源分配器 + 控制程序”	既调度硬件资源，又防止程序彼此干扰
核心目标	方便用户、高效利用硬件	用户追求易用，系统追求效率与公平
计算机四大组成	硬件、OS、系统/应用程序、用户	OS 夹在硬件与上层程序之间，起承上启下作用
中断驱动	硬件/软件事件 → 中断服务例程	OS 保存现场、分类处理，保障实时响应
双模式运行	用户态 ↔ 内核态	特权指令仅内核可执行，系统调用触发态切换
进程管理	进程=正在执行的程序，线程=轻量子任务	并发通过 CPU 复用实现，终止需回收资源
内存管理	决定“何时/何物”在内存	跟踪分配、换入换出、虚拟内存
存储管理	文件系统抽象物理介质	目录层次、映射到二级存储、备份策略
OS 服务	UI、程序执行、I/O、文件、通信、错误检测	面向用户的“便利函数库”
系统级职能	资源分配、记账、安全保护	为并发与多用户场景提供秩序与隔离
用户接口	CLI 与 GUI 并存，触屏引入手势	Shell 可扩展；桌面/移动各有交互范式
结构模式	**层次式**：环形同心层；**微内核**：极简内核+消息；**模块化**：按需装载	结构越模块化，越易扩展、移植与安全

一、操作系统基础

1. **定义与目标**
   - OS 是硬件与用户程序间的"管理者与守门员"，负责**资源分配**与**运行控制**。
   - 终极追求：**让用户方便、让硬件高效**。
2. **系统组成**
   - 四层从下到上：**硬件 → OS → 系统/应用程序 → 用户**。
3. **中断与双模式**
   - **中断机制**：CPU 收到事件→保存上下文→转入对应服务例程。
   - **用户态/内核态**：系统调用由用户态进入内核，执行完再返回。

二、核心管理功能

功能	要点
进程管理	创建/调度/终止；单线程 vs 多线程并发
内存管理	分配、置换、虚存；兼顾吞吐与响应
存储管理	文件系统抽象、目录组织、备份策略
I/O 子系统	缓冲、缓存、假脱机 + 统一设备驱动
保护与安全	访问控制列表、认证、防御外部攻击

三、操作系统服务与接口

1. **用户级服务**：UI（CLI/GUI/触控）、程序执行、I/O、文件操作、通信、错误检测。
2. **系统级服务**：资源分配、记账、安全与保护。
3. **多样化接口**
   - **CLI**：脚本友好，可扩展 Shell。
   - **GUI**：图标+指针；Mac Aqua、Windows、GNOME 等。
   - **触屏**：基于手势与虚拟键盘。

四、系统调用与系统程序

- **系统调用六大类**：进程控制、文件管理、设备管理、信息维护、通信、保护。
  (详见 slides "Types of System Calls")
- **参数传递**：寄存器、内存表、栈三种策略并存。
- **系统程序**：为开发与运维提供"第二层 API"，包括文件工具、编译器、调试器、后台服务等。

五、操作系统结构模式

结构	特征	优劣
层次式	逐层只依赖下层接口	清晰、易验证，但层间通信可能开销大
微内核	只保留最小核心（IPC、调度、内存）	可移植、安全；但用户态↔内核态通信成本高
模块化	核心功能按需装载	灵活、面向对象，现代 Linux/Solaris 通用

六、设计与实现原则

1. **策略 (Policy) 与机制 (Mechanism) 分离** —— 方便后期替换策略。
2. **实现语言分层** —— 核心汇编/C，系统程序 C/C++/脚本；高层语言利于可移植，但需权衡性能。
3. **启动过程** —— 固件加载引导块→引导装载程序→内核就绪（如 GRUB 双阶段）。

如何使用这些笔记

- **复习顺序**：先掌握“定位—功能—结构—接口”，再扩展到具体实现与设计原则。
- **快速答题**：遇到 OS 结构类问题，可先用层次式/微内核/模块化三分法组织答案。
- **深入阅读**：想看原理细节时，对照页码快速定位相应幻灯片。