



华南师范大学  
South China Normal University

## SCNU-OS-Labs

**proj0-适合本校特点的实验指导教程**

项目成员

梁铭皓

刘逸珑

赵秀明

指导老师

李丁丁

## 一、项目背景及意义

智能设备如同现代社会的新生命，而操作系统则在这个比喻中扮演着灵魂的角色。其重要性显而易见。操作系统被形容为我国基础软件国产化攻坚战中不可或缺的关键一环，如同一个巧妙调度交通的枢纽，连接着硬件和软件设施，在生态建设中发挥着至关重要的作用。

首先，操作系统负责管理计算机的硬件资源，如处理器、内存、存储设备和输入输出设备等。它有效地分配和调度这些资源，使得多个程序可以同时运行并共享计算机的资源。操作系统的优化和管理能力直接影响到计算机的性能和效率。

其次，操作系统提供了一组关键的抽象和接口，使得应用程序可以方便地访问和使用底层硬件。通过操作系统提供的文件系统、网络协议栈、设备驱动程序等，开发人员能够更轻松地编写应用程序，而无需关注底层硬件的具体细节。这种抽象层提高了开发效率并促进了软件生态系统的繁荣发展。

此外，操作系统是实现并发和并行计算的关键。在现代计算机系统中，同时执行多个任务已经成为常态。操作系统通过进程管理、调度算法和同步机制等手段，使得多个任务能够有效地共享计算机资源并协调彼此的执行。对于计算机学生来说，理解并发编程的原理和技术是必不可少的，而操作系统正是提供了相关的概念和工具。

并且，操作系统还提供了强大的安全性和保护机制，确保计算机系统的稳定性和可靠性。操作系统通过访问控制、用户身份验证、内存保护等手段，保护计算机免受恶意软件、非法访问和系统故障的影响。对于计算机学生来说，了解操作系统的安全性特性和漏洞修复是必备的知识，以确保计算机系统的安全性。

最后，操作系统的知识对于计算机科学和软件工程领域的学习和职业发展至关重要。无论是开发应用程序、设计网络系统还是进行系统维护和故障排除，都需要对操作系统的原理和机制有深入的了解。操作系统的知识可以帮助学生更好地理解 and 利用计算机系统，从而成为高效的计算机专业人员。也正因如此，操作系统对于计算机专业的学生来说也一直是不可或缺的课程，起着十分重要的作用。

与此同时，习近平总书记在党的二十大报告中所强调：“科技兴则民族兴，人才强则国家强。”这一思想在操作系统教学中得以充分体现。当今国际形势日益紧密，操作系统教学的重要性远不仅限于传授知识。它在科技创新、人才培养、信息安全、产业发展以及国际影响力等多个领域都有深远的影响。

操作系统的教学不仅培养了学生对计算机系统底层原理的深刻理解，更在科技创新中发挥着关键作用。这些知识储备不仅为技术创新提供了坚实基础，还为培养具备战略眼光的人才奠定了基础。在信息时代，信息安全至关重要。操作系统教育在这一领域发挥着保护国家信息资产和维护国家安全的使命。

同时，操作系统教育也是国家产业发展的支撑。优秀的操作系统人才能在多个行业中施展所长，推动产业升级和发展。此外，这一教育也为国际交流合作提供了坚实基础，增强了国家在科技领域的影响力。

近年来，许多国内外高校在操作系统课程实验设计上都进行了一系列的改革，MIT 的 xv6、清华大学的 ucore、南京大学的 Nanos，还有北京大学，中山大学等高校都改革了操作系统的实验环节，他们都结合本校特点改革出适合自己的实验指导教程。但每个学校的情况不一，不同高校 OS 课程的要求是不同的。

我校于 2022 年秋季时，便对 OS 课程实验进行了一次较大的变革，往年的实验内容以 Linux 内核为基础实验平台，内容几乎由课程老师自创，包括：“Linux 内核编译”、“Linux 模块编程”、“进程通信”和“进程/线程互斥锁”等内容，不失开

放性和趣味性，但是由于 OS 内核过于复杂，开发难度大，所以设置的实验总体难度偏低，在一定程度上不利于学员们学习和掌握操作系统的知识，甚至还会让学员对开发产生恐惧的心理。

改革之后，我们便以 MIT 著名的 6.828 课程为蓝本，加以一系列适合本校的改良之后，让学生不再以 Linux 这样复杂的通用 OS 内核为实验平台，而是用 xv6 这样简洁的教学型内核取代之，从而降低大家对于 OS 代码学习和开发的恐惧心理，同时增加了编程量，将大家的主要精力从“熟悉 Linux 系统”，转移至“实现 OS 所必须的基础模块和应用层工具”上。

与此同时，在过去几十年的时间里，处理器指令集一直由公司垄断，例如 ARM 指令集。即使其他企业具备了芯片微架构设计的能力，也必须获得 ARM 的授权才能研发兼容 ARM 指令集的芯片。然而，RISC-V 的崭露头角打破了这种局面。RISC-V 的出现标志着指令集不再受限于某个特定公司，而是成为全球共同建设的开放领域。这形成了一种新的处理器生态构建模式，即开源、开放、共享。

进入到 2023 年初，RISC-V 被列入《MIT 技术评论》评选的“十大突破性技术”之一，这凸显了其在芯片设计领域的重要性。这标志着新一轮的技术与产业变革浪潮正在逐步展开。选择基于 RISC-V 的处理器作为 xv6 操作系统教学工具，是出于对时代发展潮流的迎合。

通过基于 RISC-V 的处理器来教授操作系统课程，有助于让学生紧跟科技的最前沿。学生不仅可以深入了解操作系统的内部机制，还能理解处理器指令集与硬件架构的关系。这种教学方法有助于培养具备创新思维和适应快速变化技术环境能力的人才。同时，选择 RISC-V 也体现了鼓励学生积极参与开源社区、贡献代码和共享知识的价值观，为未来的科技创新和产业发展奠定基础。

但是，要想设计出一款适合本校特色的 OS 实验指导教程并不是一朝一夕可以完成的事情，经过对第一年的观察与得到的反馈，我们发现还有很多的地方可以完善。通过发放问卷得到普查回来的数据显示，第一轮改革后，虽然内核代码小了很多，但是由于所做的实验都是独立分开的，而且前期也没有相关知识的引导性学习，所以很多同学反馈上手比较难，而且体系难以建立。再者大部分同学在完成实验时，由于指导书提供的指导不够充分，很多同学产生了畏难心理，并且只有极少部分同学是独立完成了所有实验，这并不利于同学们通过该实验项目理解操作系统和培养系统编程的能力。

此外我们还对即将在下学期修读本实验项目课程的低年级同学进行了调查，发现大部分同学对于完成本实验的前置知识是大量缺失的。

根据本校实际需求，我们将继续改革完善，针对前置知识缺失的情况，我们撰写相关文档供同学们学习，也会在实验过程中设计实验题逐步引导同学们掌握所需的工具。针对操作系统实验本身，前期以引导性的方式逐步引导学生了解 xv6 内核，而且对有相关关联性的实验内容进行合并，让同学更加清晰地建立 OS 内部体系，最重要的是我们会在每个实验都有 GDB 调试的引导和问题，让同学们在真正的操作系统内核代码中亲自动手实验并逐步调试理解，从而深入理解操作系统的原理。

操作系统实验，不仅是知识的传承，更是思想的碰撞，创新的孵化场。我们鼓励学生不仅要理解操作系统的原理，更要拥有实际操作的能力，能够从代码中感受知识的力量。在这个过程中，他们将培养出对技术的热情，对问题的追根溯源，对创新的无限渴望。在这次课程改革中，我们追求的不仅是课本知识，更是跳脱束缚的思维，是能够直面挑战的勇气，是解决问题的毅力和创意。

## 二、实验设计

### 2.1 实验设计基本原则

针对本校调研后的情况，我们决定实验的设计重视基础，由浅入深，从基础知识和基础工具开始，手把手带领同学们进行实验，同时在实验过程中，添加扩展阅读，涉及一些方法论和学术前沿的知识，丰富同学们的知识面。此外对于基础较好的部分同学，也提供了附加的选做题，提升该部分同学的能力。

在实验设计中，尽可能做到对本校绝大部分同学的覆盖，使得基础一般的同学能够通过实验项目迅速掌握系统编程的基本能力，而基础较好的同学也可以在继续锻炼编程能力的同时，接触到一些扩展知识，培养在操作系统领域的兴趣，为日后的科研工作打好基础。

### 2.2 前置知识补充

在针对即将参与此实验的低年级同学的问卷中，我们了解到了同学们对于前置知识的一些缺失，所以我们决定增加部分前置知识补充。

对于操作系统实验来说，Linux 系统相关的内容是不可缺少的，我们在开始实验设计之前进行了一项摸底调查，旨在了解学员对 Linux 系统的了解程度。在针对是否使用过 Linux 系统的问题调查中，我们发现 56% 的同学表示完全没用过 Linux 系统，也有 28% 的同学表示有接触过但是并不熟悉，能熟悉运用 Linux 系统的人数占比相对来说是最少的，仅有 16%。针对此情况，我们特意在指导书中加多了 Linux 使用基础教程，方便同学们可以快速上手一些基本的知识。

首先我们针对未曾使用过 Linux 系统的零基础的同学设置了“Linux 安装与基础环境搭建指引”这一板块，旨在最快速度的帮助他们能够简单的能在 Linux 操作系统中完成本实验。

我们还设置了“常用命令速查表”这一模块，让对于 Linux 系统下的命令不熟悉的同学可以迅速查找常用的命令，并且我们还提供了 Linux 系统入门教程的链接，方便需要的同学跳转学习。在该模块中，除了有 Linux 系统的基本命令和入门教程之外，还设置了 TMUX 终端复用器的基本操作命令、GDB 调试器的常用命令，还有在实验中需要用到的 QEMU 模拟器的一些基本命令。

## 常用命令速查表

### Linux系统

- `ls`：用于列出当前目录下的所有文件，`ls -l` 可以显示详细信息
- `pwd`：能够列出当前所在的目录
- `cd DIR`：可以切换到 DIR 目录，在Linux中，每个目录中都至少包含两个目录，`.` 指向该目录自身，`..` 指向它的上级目录，文件系统的根是 `/`
- `touch NEWFILE`：可以创建一个内容为空的新文件 NEWFILE，若 NEWFILE 已存在，其内容不会丢失

图 1 常用命令速查表板块

如果同学们遇到解决不了的问题，还可以查看我们提供的问题解决指南，里面提供了一些解决问题的方法论以及有用的链接，可以帮助同学们迅速定位解决问题的方向。

此外，在 xv6 实验中会涉及到 RISC-V 指令集的内容，所以我们也添加了介绍 RISC-V 的板块，以便于在计算机组成原理课中学过 MIPS 的同学们可以快速进行知识迁移。在问卷摸底调查中，我们发现超过一半同学不能读懂汇编代码，目前的解决方法是尽可能在实验指导文档中进行一些解析，后续可能会加入一个前置实验以便于同学们熟悉 RISC-V 的汇编代码。

在系统编程中，使用 GDB 来定位 bug 也是必不可少的，并且我们在尝试后发现，使用 GDB 对 xv6 进行某一个模块的过程调试可以快速理解该模块的代码动态行为，有助于理解代码框架。我们使用问卷对参与过第一次改革的上一届同学进行调查，结果超一半同学表示在实验中并没有用到 GDB。而对即将要参与实验的下一届同学，我们调查发现有 81.82% 的同学表示没有用过 GDB。大部分同学在 debug 时都是使用“瞪眼法”或者“print 大法”。

于是我们在 Lab0 中设置了与 xv6 无关的实验，以便于让同学们初步接触在 Linux 环境下编程和使用 GDB。

#### 学习使用gdb

使用gdb调试你的“Hello World”程序，在主函数处打断点，并单步调试观察主函数的行为，此外你还可以在该程序中探索更多gdb的功能。

#### 别偷懒！

上面叫你写一个“Hello World”程序并且学习一下gdb，可不要偷懒啊，特别是此前还没有接触过Linux下编程和gdb调试的同学。

如果是此前已经具有Linux下编程经验的同学，你当然也可以选择性跳过上面的任务，选择下面的思考题，我们也非常欢迎其他同学在完成上述任务后选择做下面的思考题。

#### 调试一个Segmentation fault的小程序

将下面两个文件中的代码放置于同一文件夹下，输入 `make` 指令，GNU `make` 会根据 Makefile 中的依赖和构建关系自动生成 `a.out` 文件。

运行 `a.out` 文件，你会发现出现了 `Segmentation fault`，那么为什么会出现 `Segmentation fault`？

使用gdb从第一条指令开始执行，观察寄存器值的变化，你会找到答案的。既然出现了 `Segmentation fault`，要如何修改代码才能使程序正常运行呢？把答案写在你的报告中。

图 2 Lab0 中使用 GDB 的实验

## 2.3 基本框架

对于整个实验的基本框架，我们打算设计如下几个实验：环境配置相关、xv6 代码结构的导读和内核启动的初步调试、系统调用、内存管理、进程调度、文件系统。在全国赛阶段 1 结束前，我们完成了 Lab0 到 Lab5 六个实验的设计，基本完成了我们的计划。

### 2.3.1 Lab0 环境配置

在该实验中，我们会让同学们首先进行简单的 Linux 环境下的编程和 GDB 调试的学习，如图 2 所示。当然，就目前来看该实验设计的略简单，在后续可能会结合 RISC-V 的汇编，设计一个类似于 CSAPP 中 Bomb 实验的 RISC-V 版作为前置实验，

让同学们在调试过程中熟悉 GDB 使用和 RISC-V 的汇编代码。

然后手把手带领同学们在不同的系统下配置实验的基本环境，获取实验代码，最后以把 xv6 跑起来作为本实验的结束。针对 macOS 的环境配置，在全国赛阶段我们真实使用 macOS 重新进行了尝试，设计出了较为方便的配置方案。

### 2.3.2 Lab1 初探 xv6

针对上一届同学问卷调查的结果反馈，我们设置了一个 xv6 代码初探的实验，以方便同学们在没有接触过 xv6 的情况下，对其有一个大致的了解，方便后续实验的进行。

首先以代码树的形式把 xv6 中关键的代码文件做了介绍。然后教同学们使用 make 工具打印命令执行的过程，了解到 xv6 中大量的文件是以一个怎么样的顺序逐步执行的，接着使用 GDB 从启动 xv6 的第一条指令开始，观察整个过程中执行了哪些函数，寄存器发生了什么变化。在这个过程中，除了让同学们对 xv6 启动部分的代码有所了解之外，还让同学们接触到了一些工具如 make 和 GDB 的使用，了解到这些工具对于提高系统编程解决问题的效率有什么帮助。并且也作为一个例子，教会同学们怎么用 GDB 调试 xv6 的一小块代码。

最后，增加一个趣味性的实验，让同学们使用工具自行定位到 xv6 中设置 shell 的代码所在位置，修改代码，定制自己的 shell。

### 2.3.3 Lab2 系统调用

在本实验中，针对上一届同学在进行系统调用实验时反馈的实验指导书并不能帮助理解整个过程，且涉及到了后面出现的 trap 部分内容，大部分同学还需要花时间自行寻找相关资料学习后才能动手。于是我们决定综合 MIT 课程实验中相关部分，将几个相关实验结合起来，更有助于同学们理解这个过程，并且更好地上手实验。

我们先让同学们使用系统调用写一些用户程序，理解系统调用在用户程序中发挥的作用。然后会结合源码，图文结合，介绍系统调用的整个过程。由于仅仅阅读指导书和简单阅读源码未必能很好理解这个过程以完成后续实验，我们设计了 GDB 调试相关的小实验题，让同学们从用户空间开始，跳转到内核空间，逐步调试观察整个过程中相关寄存器的变化，以及关键函数是怎么执行的。

有了前面的前置知识，接下来我们会结合“打开程序执行，跟踪系统调用”这个话题介绍 strace 工具，并且让同学们在 xv6 中实现获取内核执行信息的工具型的系统调用。

### 2.3.4 Lab3 内存管理

对于内存管理的实验，首先在指导书中介绍了 RISC-V 中的 SV39 的硬件机制和三级页表的实现，由于此前同学们并没有在课堂上正式学习过 RISC-V，所以在本次实验中尽可能以简单的方式介绍最基本的、实验中必要的 RISC-V 的相关知识，以便于同学们可以完成实验。

针对于内存管理的分页机制，会介绍 xv6 中的页表实现，让同学们通过 xv6 中的实例感受真实操作系统中的内存管理机制。并通过完成一个打印页表的工具深刻理解其中的原理。

之后由页表引向缺页异常这个知识点，缺页异常在本校往年的实验中并没有出现。在此处添加该知识点，是因为这是一个与内存管理息息相关且无法避免的知识，而且结合前面的实验中系统调用的知识，同学们可以很快上手这个实验，体现了一定的综合性。此外，缺页异常的实验中会天然地引入一些 bug，对提升同学们在系统编程中的 debug 能力很有帮助。

### 2.3.5 Lab4 进程调度

在本校往年的实验中，并没有出现进程调度相关的内容，但是作为操作系统中重要的组成部分，没有该部分实验也不合理，于是本次实验中加入一个进程调度的实验版块。

但是，在 xv6 中，涉及到该部分的代码相当多且复杂，还有一些硬件中断问题和并发问题相关的代码。RISC-V 硬件方面的知识同学们所知甚少，而并发问题则是一个非常复杂的问题，此前同学们对并发问题的编程基础极少，此处贸然展开并不妥当。于是，该实验中只是简单的介绍了 xv6 中内核线程的轮询调度机制，然后设计一个会发生进程调度的程序，通过 GDB 调试的方法让同学们了解进程调度的过程，打印相关进程信息，知道在何时完成切换。最后再让同学们参考内核线程的实现自己实现用户线程的调度机制。对于学有余力的同学，也添加了扩展实验部分，鼓励为系统增加更多的调度策略。

### 2.3.6 Lab5 文件系统

鉴于本校在操作系统理论课时文件系统部分讲授较少，大部分同学对于文件系统的了解不多，理解不深。于是在该部分，我们结合 xv6 中的文件系统源码，撰写了近万字的实验指导书，帮助同学们了解文件系统，除此之外，还添加了详细教材的扩展阅读，以便于同学们查阅学习该部分理论知识。

在具体实验方面，主要设计了实现文件系统二级索引和实现软链接两部分，并且做了较为详细的实验指导。此外还设计了对文件操作相关系统调用的 GDB 调试的实验。

## 2.4 扩展部分

由于实验需要覆盖大部分同学，根据调查结果，实验主体部分设置的较为基础。针对有一定基础的同学，我们设置了扩展的选做题，旨在让该部分同学可以在实验中有所提高。对于基础一般的同学，也可以在完成基础部分后继续完成扩展部分，逐步提升自己的系统编程能力。后续还会考虑设置结合多个模块知识的综合性的扩展实验，合理设置三个难度梯度，以更好地覆盖不同基础的同学。

目前设置的扩展实验主要有：定制功能更完善的 shell，给 xv6 中 find 命令工具添加对正则表达式的支持，在 trace 工具中打印所跟踪的系统调用的参数，调试更多文件操作的系统调用等。这些扩展实验大多与现代的 Linux 系统中的工具息息相关，在实现扩展实验的过程中，参考 Linux 系统的各种命令的帮助手册是必不可少的，通过扩展实验可以使同学们对现代的 Linux 系统了解更多，对于自己想操作系统中实现的功能，也可以参考 Linux 的实现。

除了扩展实验，实验指导文档中还会有针对每个实验的扩展阅读，在相应的地方帮助同学们扩展知识面。我们扩展阅读的知识包括一些操作系统重要的技术和思想、历史上与操作系统相关的有趣的案例。

## 2.5 实验测试

目前，我们使用的是 Python 脚本进行本地测试的方法，针对每一个编程实验，我们设置了一些正确的预设结果，通过执行相应的指令，同学们可以测试代码执行的结果是否正确，还可以进行整个实验的全面测试，看到自己该实验下拿到的编程练习的分数。目前，选做的编程练习还没有设置测试。后续还考虑加入自动测试功能，将自动测试与评分结合起来，有助于教师端进行打分。

```

== Test trace 32 grep == trace 32 grep: FAIL (5.6s)
...
    hart 2 starting
    init: starting sh
    $ trace 32 grep hello README
    exec trace failed
    $ qemu-system-riscv64: terminating on signal 15 from pid 419 (make)
    MISSING '^d+: syscall read -> 1023'
    QEMU output saved to xv6.out.trace_32_grep
== Test trace all grep == trace all grep: FAIL (0.7s)
...
    hart 2 starting
    init: starting sh
    $ trace 2147483647 grep hello README
    exec trace failed
    $ qemu-system-riscv64: terminating on signal 15 from pid 800 (make)
    MISSING '^d+: syscall trace -> 0'
    QEMU output saved to xv6.out.trace_all_grep

```

图 3 测试结果示意图

### 三、工作总结和未来展望

#### 3.1 工作总结

截止于全国赛阶段 1 结束，我们项目的完成情况如下：

表 1 项目完成情况

目标	完成进度	说明
设计，发放并分析问卷	100%	这是针对我校实际情况设计实验指导教程的基础，问卷中着重了解学生们一些基础知识情况以及对 OS 理论课的一些反馈。
改进实验指导文档	100%	参考国内外各大学的 OS 课程教学指导，再结合本校特色，设计出不同难度的实验题目，让学生在一步步实验中了解 OS 的相关机制。
部署实验代码框架	100%	已在 GitLab 项目仓库部署现已完成的实验代码框架。
设计实验测试用例	100%	为每个实验提供测试方法，指导学生对所做实验进行相应测试。

我们目前完成计划开发的实验有 Lab0 环境配置、Lab1 初探 xv6、Lab2 系统调用、Lab3 内存管理、Lab4 进程调度、Lab5 文件系统。这几个实验主要是分开不同的实验代码分支进行的，后续考虑将其结合起来，形成在一个操作系统上不断完善功能的布局。

对于实验指导文档，除了每个 Lab 相应的指导书，我们还在导读部分设置了前置知识的板块。

总体来说，我们目前完成了：

- 本校实验情况调研
- 整个实验项目的实验指导文档

- Lab0-Lab5 的实验代码部署
- 现有实验的代码测试脚本
- 针对不同基础同学划分基本和扩展部分



图 4 实验指导文档展示

Name	Last commit	Last update
conf	update lab into 2	6 days ago
kernel	syscall lab	1 year ago
mkfs	die	1 year ago
user	update lab into 2	6 days ago
.dir-locals.el	Setting indent-tabs-mode nil everywhere is dangerous	11 years ago
.editorconfig	Clean up linker script	3 years ago
.gdbinit.tmpl-riscv	set riscv use-compressed-breakpoints yes	2 years ago
.gitignore	syscall lab	1 year ago
LICENSE	x	3 years ago
Makefile	Makefile: add diff and tarball for Submit work	6 days ago
README	update	1 year ago
grade-lab-2	update grade-lab-2	13 hours ago
gradelib.py	syscall lab	1 year ago

图 5 代码部署展示

### 3.2 创新点

在每个实验中，针对该实验的板块，我们进行了对该板块源代码进行 GDB 逐步调试的教学和设置实验操作练习，使得同学们在整个过程中，一边提升 GDB 调试能力，一边熟悉代码框架。

在实验过程中，设计基础实验题目、基于基础实验题目的进阶扩展题、综合不同基础实验题目的综合性实验，实现基础-进阶-拔高三个层次的逐步结合，既有利于有基础的同学迅速上手难度较高的实验，也有利于基础一般的同学从基础题一直按部就班做到拔高题。

结合了计算机系统基础、操作系统的实验，结合了简单的 xv6 操作系统和真实 Linux 系统环境的实验。在实验设计中，以 xv6 为主线，穿插系统基础相关的知识和练习，也结合了 xv6 相应模块下一些 Linux 下相关的工具等设计了一些小 demo，结合 Linux 相关内容设计 xv6 内的实验，以便于同学们从简单的 xv6 中学习操作系统各个模块的组成和实现，同时接触到真实环境中操作系统是怎么工作的。

### 3.3 未来展望

进一步优化实验指导文档的结构和内容，增加有用的扩展阅读部分。  
进一步完善测试用例和脚本，将基础必做题和进阶选做题的测试打分分离。  
将多个代码分离的实验板块结合，使同学们在一个操作系统下进行功能完善。  
进一步完善选做题内容，结合多个模块设计综合性实验。  
继续增加 GDB 调试相关部分，将其引入编程练习题之中。  
增加可视化 demo，帮助同学们理解对应的知识点。  
设计 RISC-V 版本的 Bomb 实验，从中同时学习 GDB 调试和 RISC-V 汇编代码。  
继续增加 xv6 和 Linux 结合部分，将实验和真实环境结合。  
将实验部署到 GitHub Classroom，完成自动测试评分功能。

## 四、参考文献

- [1] F Kaashoek, R Morris. MIT6.S081. <https://pdos.csail.mit.edu/6.S081/2021>. 2021
- [2] 蒋炎岩. 操作系统：设计与实现. <http://jyywiki.cn/OS/2023>. 2023
- [3] 陈海波，夏虞斌. 操作系统：原理与实现. [ipads.se.sjtu.edu.cn/osp/2023](http://ipads.se.sjtu.edu.cn/osp/2023). 2023
- [4] 余子濠. 南京大学计算机系统基础课程实验. <https://nju-projectn.github.io/ics-pa-gitbook/ics2022>. 2022