Lab1 System Calls 实习说明

本 lab 的任务是添加系统调用，理解 xv6 内核和系统调用。

详细要求及提示见链接：
(https://pdos.csail.mit.edu/6.828/2020/labs/syscall.html)

实习内容

Exercise 0 源代码阅读

阅读下列源代码，理解 xv6 内核和系统调用。

- `user/user.h`, `user/usys.pl`
- `kernel/syscall.h`, `kernel/sysproc.c`
- `kernel/proc.h`, `kernel/proc.c`

Exercise 1 System call tracing

添加系统调用 `trace`，该功能在调试以后的 lab 时会有所帮助。

系统调用接收一个参数，即整数“掩码”，指定要跟踪的系统调用。例如，要跟踪 `fork` 系统调用，程序将调用 `trace(1 << SYS_fork)`，其中 `SYS_fork` 是 `kernel/syscall.h` 中的 syscall 号。

修改 xv6 内核，使得如果一个系统调用在掩码中被设置，则必须在每次系统调用即将返回时打印出一行。 该行应包含进程 ID，系统调用的名称和返回值；无需打印系统调用参数。`trace` 系统调用应启用对调用它的进程及其随后派生的所有子进程的跟踪，但不应影响其他进程。

Exercise 2 Sysinfo

添加系统调用 `sysinfo`，收集正在运行的系统的相关信息。

系统调用接收一个参数：一个指向 `struct sysinfo` 的指针（请参阅 `kernel/sysinfo.h`）。内核应填写此结构体的字段：`freemem` 字段应设置为可用内存的字节数，`nproc` 字段应设置为 `state` 为 `UNUSED` 的进程数。