Lab 09：进程管理与编程实践

1. 实验目标

- 掌握Linux中与进程相关的常用命令。
- 熟悉使用系统调用创建和管理进程。
- 能够使用 `execve` 系统调用执行程序。
- 理解父子进程关系及进程同步机制。

---

2. 实验内容与要求

2.1 课堂练习：进程相关命令实践

任务一：使用命令查看进程信息

阅读对应命令的 `man` 手册，掌握常用开关选项。**文档中需简要描述所使用的开关及观察到的信息。**

1. 使用 `ps` 命令快速查看进程列表：
    - (a) 只列出当前用户的进程。
    - (b) 列出所有用户的进程，并显示每个进程对应的用户信息。
    - (c) 显示进程启动时的命令行参数信息。
2. 使用下列命令进一步观察进程状态：
    - (a) 使用 `pstree` 展示当前系统中的进程树结构。
    - (b) 使用 `pidof` 显示指定命令对应的进程ID，例如查询 `pidof` 命令本身的PID。
    - (c) 使用 `lsof` 列出当前打开的文件，并结合 `grep` 过滤结果，只显示特定程序（如 `lsof` 本身）相关的打开文件。

2.2 编程实践：进程创建与管理

编程任务一：使用 `execve` 执行命令

- 编写一个C程序，使用 `execve` 系统调用执行系统命令 `ls -l`，列出当前目录下的文件和目录。

要求说明文档中简要描述 `execve` 的使用方式及程序执行效果。

编程任务二：子进程中使用 `execve`

- 编写一个C程序：
    - 创建一个子进程。
    - 在子进程中调用 `execve` 执行系统命令 `ls -l`。

- 父进程等待子进程结束。

要求说明文档中描述父子进程的创建逻辑及观察到的执行流程。

编程任务三：父子进程管理与退出信息收集

- 编写一个C程序，要求实现：
  1. 父进程创建**三个子进程**。
  2. 每个子进程：
     - 睡眠一段**随机时间（1～9秒）**。
     - 睡眠结束后打印自身的**PID**和睡眠时间。
     - 退出时，将**睡眠秒数作为退出码**返回。
  3. 父进程：
     - 使用 `waitpid` 逐个等待子进程结束。
     - 打印每个子进程的**PID**和**退出码**。

要求说明文档中清晰说明：

- `fork`、`sleep`、`waitpid` 的使用方式。
- 父子进程如何同步以及如何获取退出码。

---

3. 提交内容

- 源代码文件（需包含必要的注释）。
- 实验说明文档（简要总结每个任务的完成情况，描述关键命令和函数的使用）。