**Operating-System Note**

**CH2、  
  
CH3、**

1、fork：linux下调用fork会返回两次，在父进程返回子进程pid，在子进程中返回0，子进程实际pid不为0，可以以此区分父进程与子进程。  
 调用fork（）后，子进程与父进程均往下执行代码。

Project3：

未知bug：将数据先写进一个字符串数组（长度大于需要写入的数据长度）中，大概率系统崩溃。直接将数据写进usr\_buf成功。应该是内存管理存在问题。（艹，题目没要求这个）

1. for\_each\_process(task\_struct \*task)，for\_each\_process是一个宏：#define for\_each\_process(p) //从init\_task开始遍历所有进程，p的类型是task\_struct指针  
    for (p = &init\_task ; (p = next\_task(p)) != &init\_task ; )
2. 可用其遍历所有进程

进程名称为task->comm,具体可看task\_struct

Project3-2

list\_for\_each()是一个宏：

#define list\_for\_each(pos, head) \  
for (pos = (head)->next; pos != (head); pos = pos->next)

就是遍历链表，传入进程的children list\_head，可以遍历下一代子进程，将其用在函数递归即可实现深度搜索。

得到的children指针使用list\_entry()函数可反推task\_struct的指针

task = list\_entry(children, struct task\_struct, sibling);

Children在struct中为sibling成员

**CH4、**

**Project1、**创建线程，join可阻塞进程