**操作系统课程设计报告**

**Project 2**

**姓名：倪浩垠**

**学号：517020910191**

**任课教师：吴晨涛**

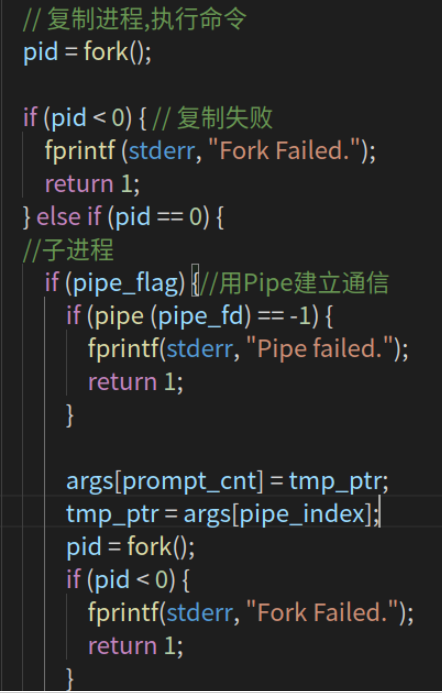
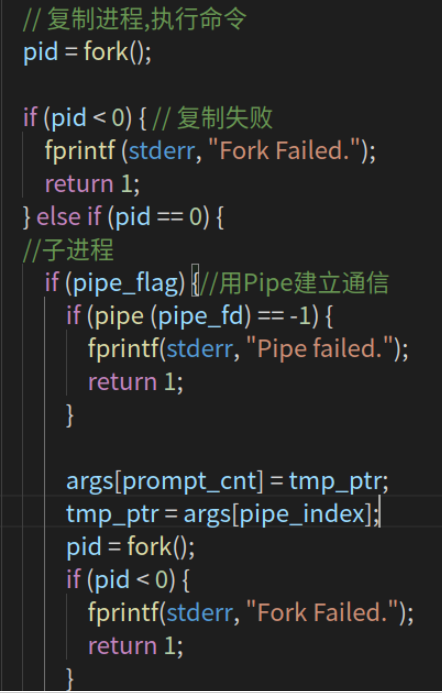
一、 Unix Shell

1 项目内容

Shell给用户提供一个提示符，在提示符后输入命令。用多线程实现Shell，用户输入命令后会创建子进程来执行命令。

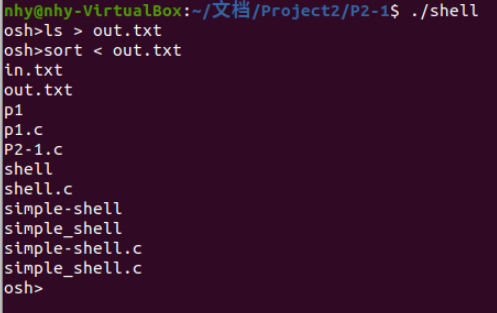
2 代码分析

用fork()系统调用来复制进程

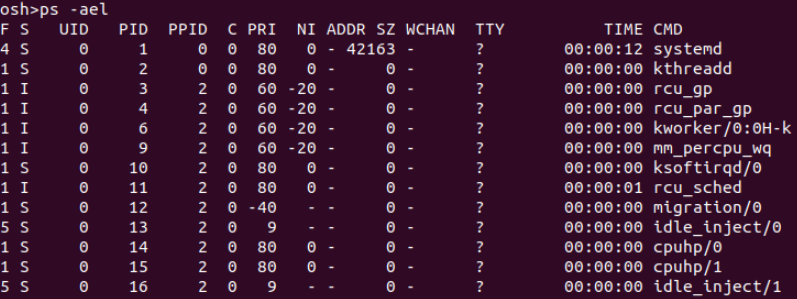


3 结果演示

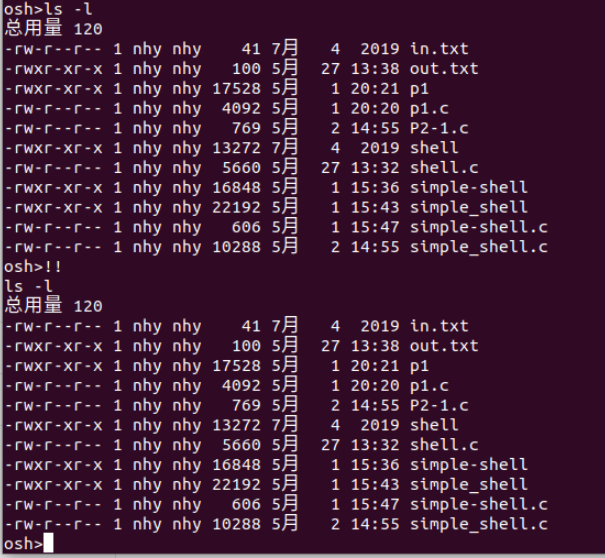
文件操作



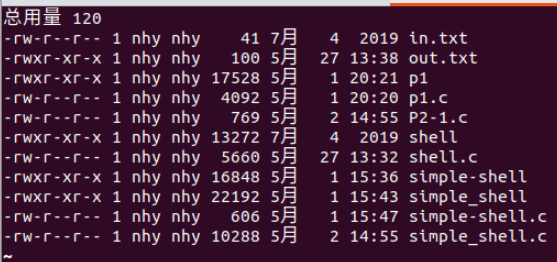
ps -ael



历史操作



ls -l | less



3 项目心得

通过这个项目，我对shell有了初步且使馆的了解

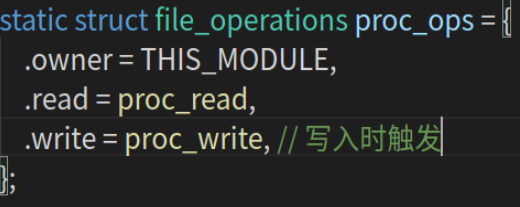
二 编写展示进程信息的内核模块

1 项目内容

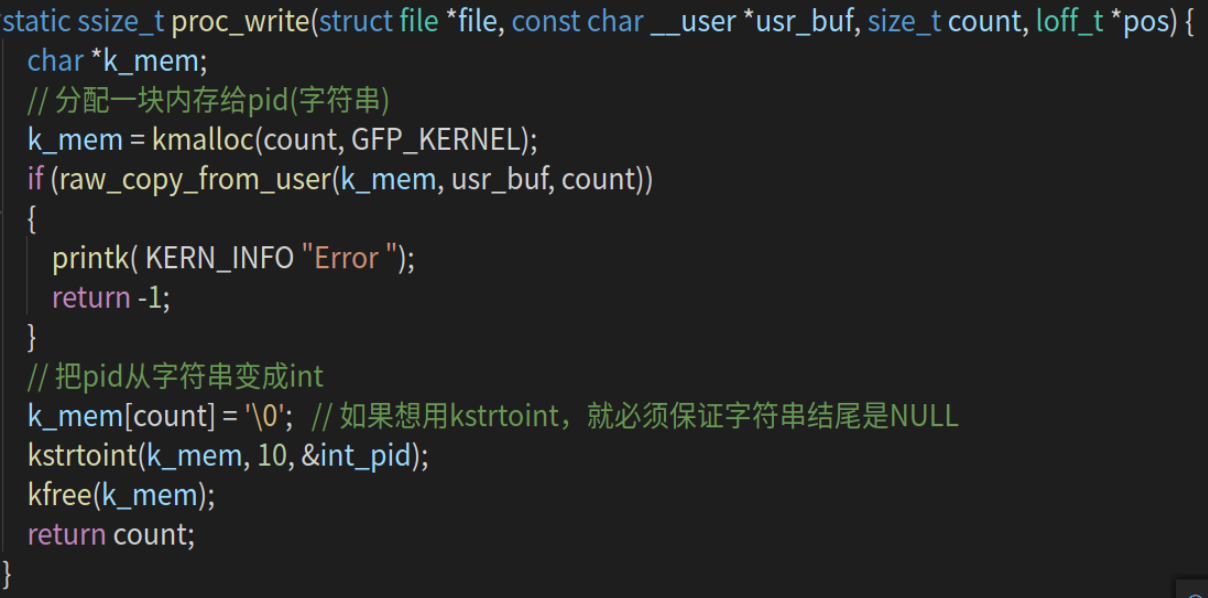
写一个内核模块。它能创建/proc/pid文件。用 echo "1395" > /proc/pid往pid文件内写入进程号为1395的进程信息。当使用cat触发proc\_read函数的时候，读取该进程的信息。

2 代码分析

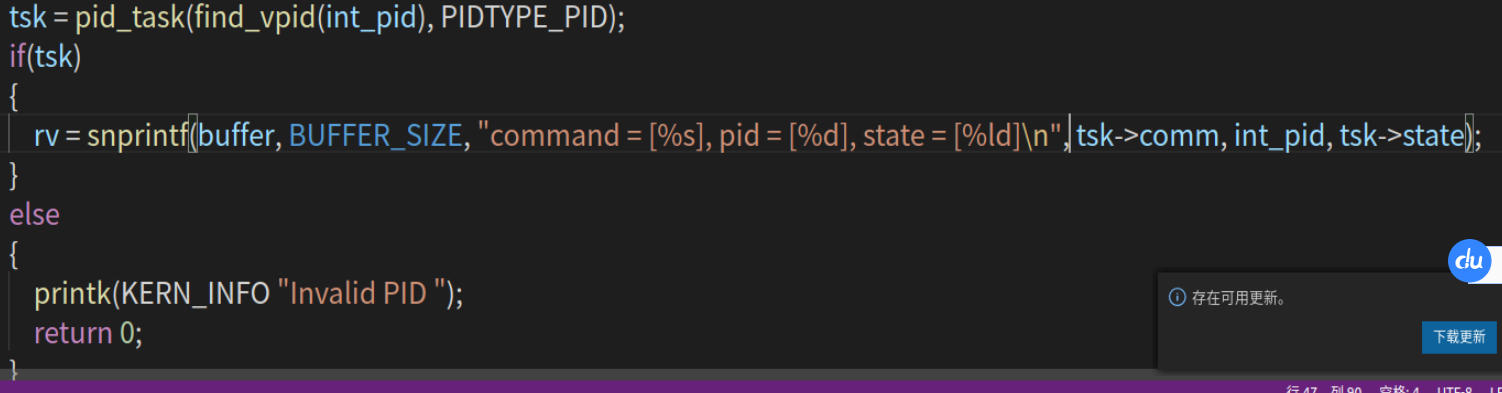
框架与Project1类似，需要添加写入时触发的函数proc\_write。



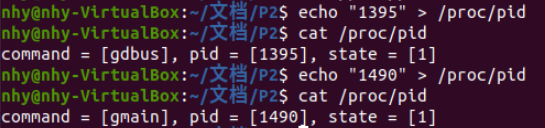
proc\_write。需要注意的是，在k\_mem中储存的是字符串，需要将它转为整形。



在对proc\_read进行修改。



3 结果演示

在命令行中输入echo “1395” > /proc/pid，然后访问该文件，能成功显示1395号进程的信息。

4 实验心得

*4.1 遇到的困难*

这个模块由于会用到pid\_task，所以一定要写上MODULE\_LICENSE(“GPL”),否则载入模块时会报错。



*4.2 收获*

我了解了task\_struct的结构。

三 用于生成任务列表的Linux内核模块

1 项目内容

* 1. *任务的线性迭代*

编写一个内核模块，以便列出Linux系统内所有的当前任务。通过宏for\_each\_process()，迭代系统内的所有任务。针对每个任务，输出command, state和pid.

* 1. *采用dfs的迭代*

采用dfs树，迭代系统内的所有任务。

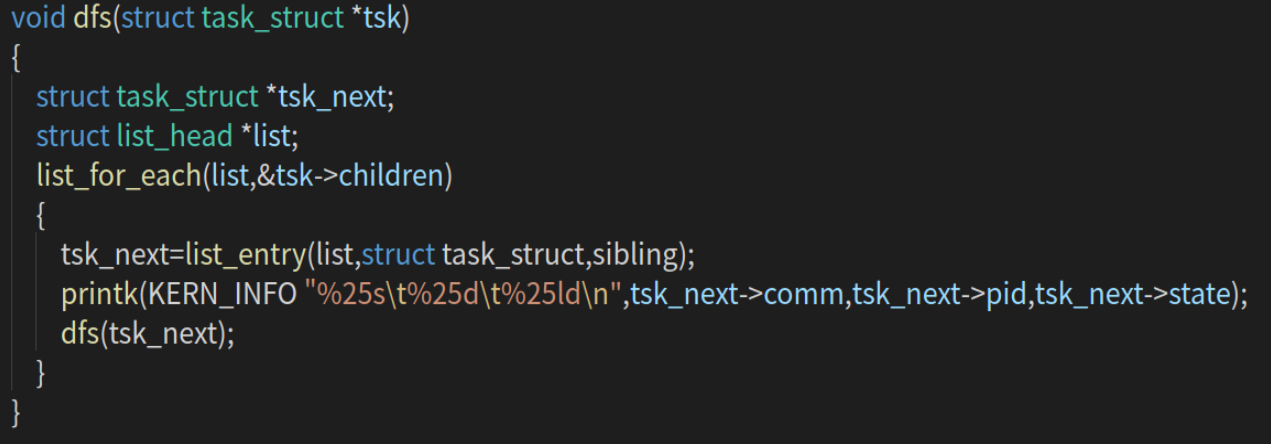
1. 代码分析

*2.1 线性迭代*

先找到第一个进程，然后用宏for\_each\_process便可以开始线性迭代。

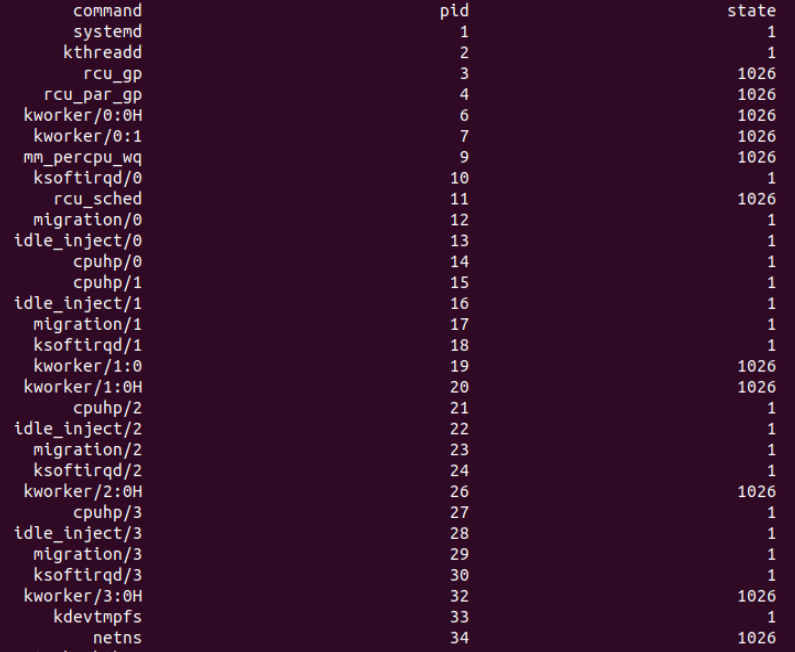


*2.2 dfs*

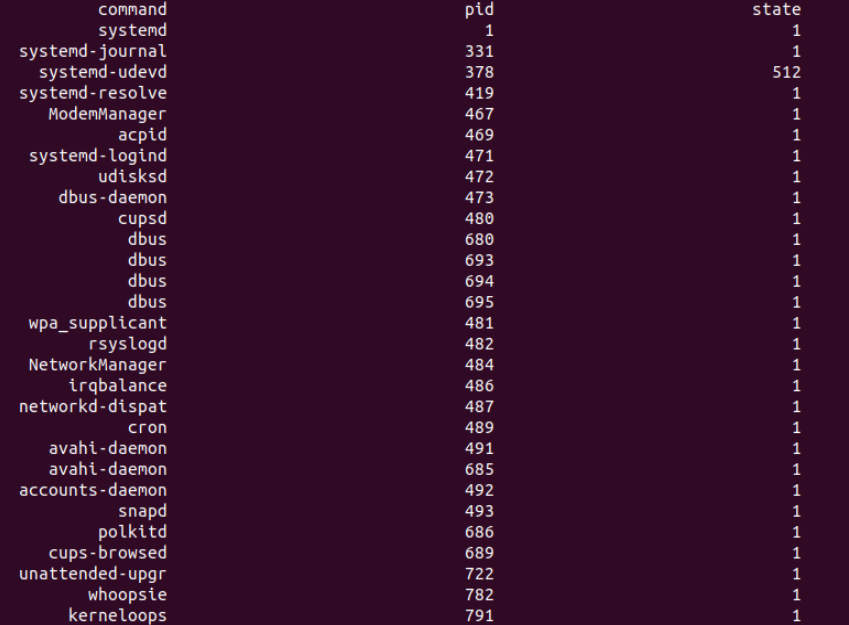


1. 结果演示

线性迭代的演示



dfs演示。可以看到，输出时的顺序与线性迭代不同。



1. 实验心得

我知道了task struct的写法，并且复习了一遍dfs。

四 内核数据结构

1 项目内容

1.1

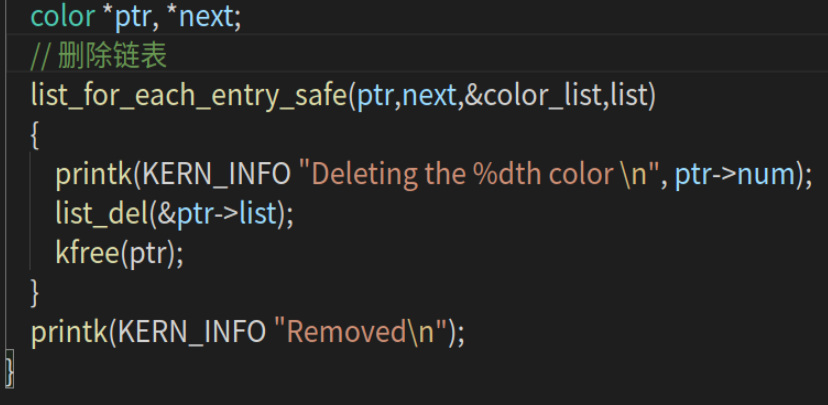
载入模块时创建一个双链表，然后遍历链表并输出。移除模块时删除链表，并释放内存。

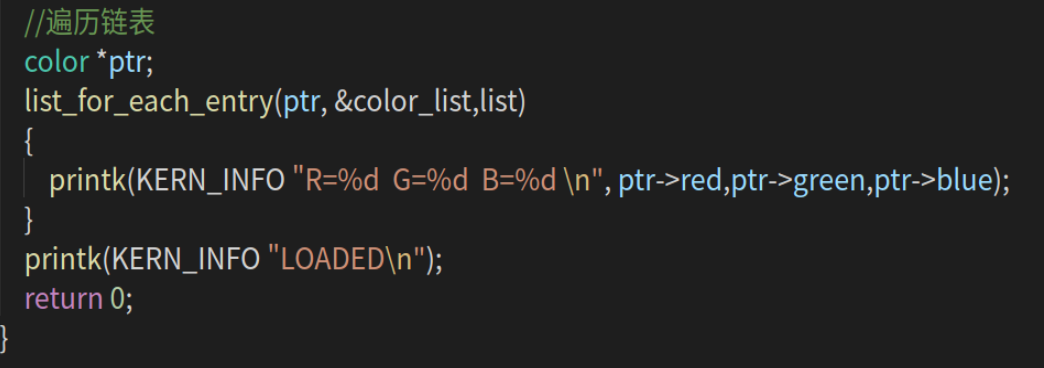
1.2

在1.1的基础上，载入模块时根据传入的参数创建collatz序列链表。

2 代码分析

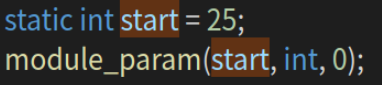
2.1 根据书上示例删除和遍历链表。List\_for\_each\_entry三个参数分别为元素结构的指针，头结点和节点中list\_head对应的变量名。





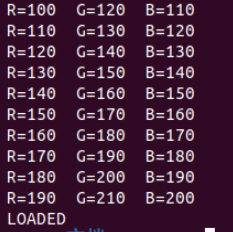
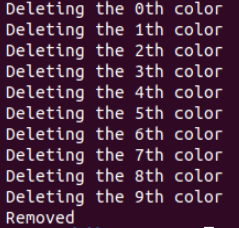
2.2

用module\_param()来传入参数。第一个参数是变量名，第二个是变量类型，第三个填0表示默认情况。



3 结果演示

载入模块时输出链表 移除模块时删除链表



首项为25的Collatz序列 首项为35的Collatz序列

