**操作系统课程设计实验报告**

——实验二：作业调度实验

负责人姓名：于福洋

学号：14061134

日期：2016.4.11

**小组成员**

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 姓名 | 学号 | 实验分工 |
| 1 | 詹子豪 | 14061137 | 实验一 |
| 2 | 于福洋 | 14061134 | 实验二 |
| 3 | 于建勋 | 14061125 | 实验三 |
| 4 | 石浩然 | 14061122 | 实验四 |

目录

[1.实验目的 4](#_Toc446001831)

[2.需求说明 4](#_Toc446001832)

[2.1基本要求 4](#_Toc446001833)

[2.2 提高要求 4](#_Toc446001834)

[2.3 完成情况 4](#_Toc446001835)

[3.设计说明 6](#_Toc446001836)

[3.1 程序流程图 6](#_Toc446001837)

[3.2基本要求实现说明 6](#_Toc446001838)

[3.3 提高要求实现说明 6](#_Toc446001839)

[4.收获和感想 24](#_Toc446001840)

# 1.实验目的

# 理解进程调度和作业调度的区别和联系。

# 学习linux系统中进程间通信的概念和方方法。

# 理解几几种常用用的调度算法,并能进行行性能分析。

# 能够理解和使用用linux给进程调度提供的系统函数。掌握进程控制和进程间通信的方法

# 2.需求说明

## 2.1基本要求

## 1. 实现作业的入队操作

## 2. 实现作业状态命令查看

## 3. 实现出队命令操作

## 2.2 提高要求

## 实现多级反馈的轮转调度算法

## 将stat 命令的就绪队列信息显示在命令终端

## 调试程序状态信息

## 2.3 完成情况

## 1. 实现作业的入入队操作

## 2. 实现作业状态命令查看

## 3. 实现出队命令操作

## 实现多级反馈的轮转调度算法

## 将 stat 命令的就绪队列信息显示在命令终端

## 实现调试程序状态信息

# 3.设计说明

## 3.1 程序流程图

创建fifo文件

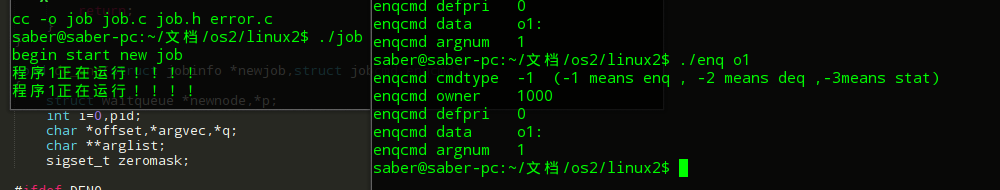
用enq stat deq 向文件中输入命令

Job从文件中读取指令进行运行

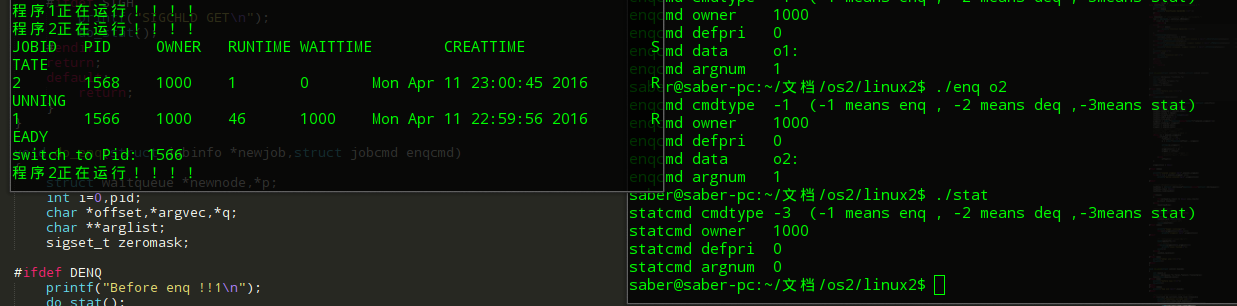
多级轮转调度

## 3.2基本要求实现说明

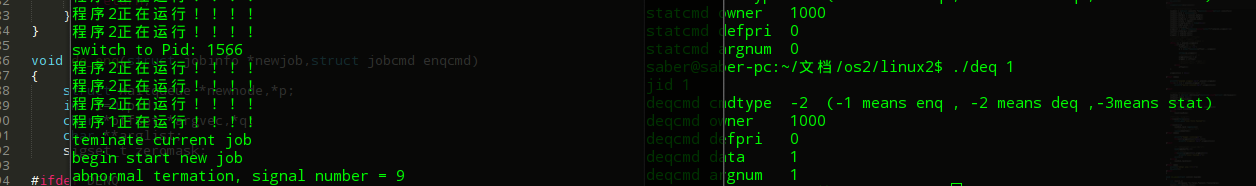
## 1. 实现作业的入队操作



## 2. 实现作业状态命令查看



## 3. 实现出队命令操作



1. 对于bug的修正

（1）fork之后父进程进行得快于子进程。父进程在执行完jobswitch()里的**kill(current->job->pid,SIGCONT)**之后，子进程才执行到**raise(SIGSTOP)**语句，这就导致了顺序的错误。因此第一个作业并不执行。而当加入第二个作业后，父进程执行了两次切换作业的语句，此时第一个作业才开始执行（在此之前第二个作业也不执行）。

在**kill(current->job->pid,SIGCONT)** 加入一句sleep（1）；就可以使自己成执行完。

（2）第二三四个错误由于要写多级轮转算法，将jobswitch和jobselect重写。再重写的过程中注意避免了这些bug，代码在下面第一个提高要求中。

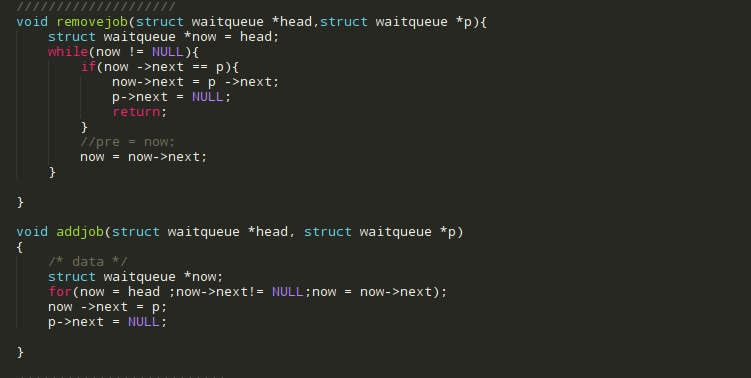
（3）Demo会导致程序在运行中的时候多跑一秒，修改为没两次之间差为1的时候才输出。

## 3.3 提高要求实现说明

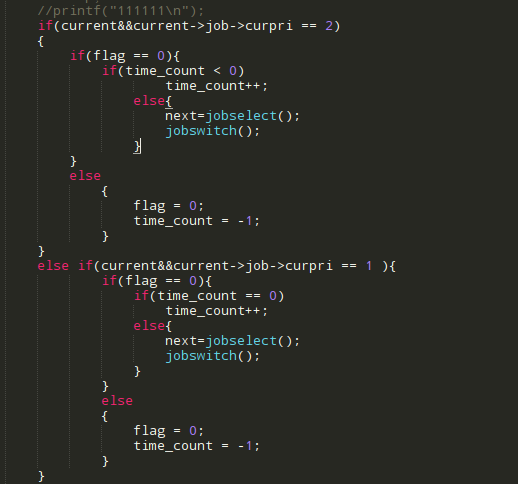
1.实现多级轮转

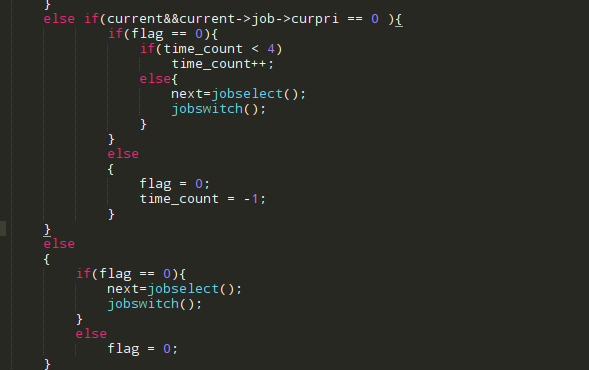
代码修改：

新增两个函数入队出队：

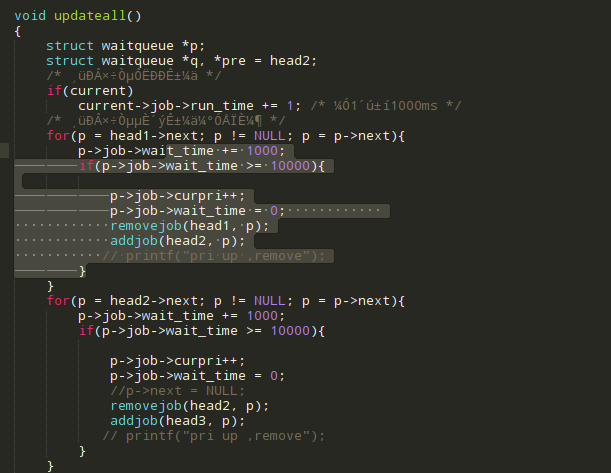


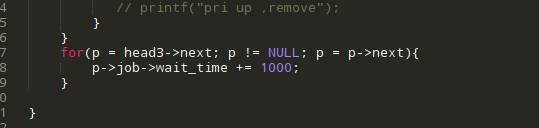
对schedule的修改。





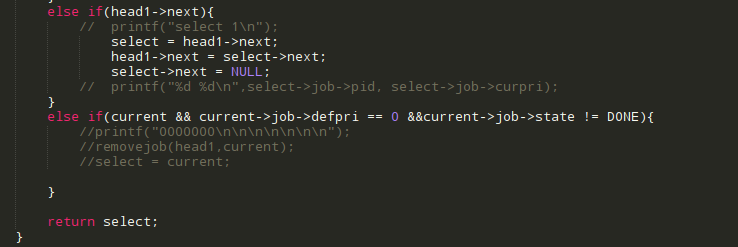
对updateall的修改。



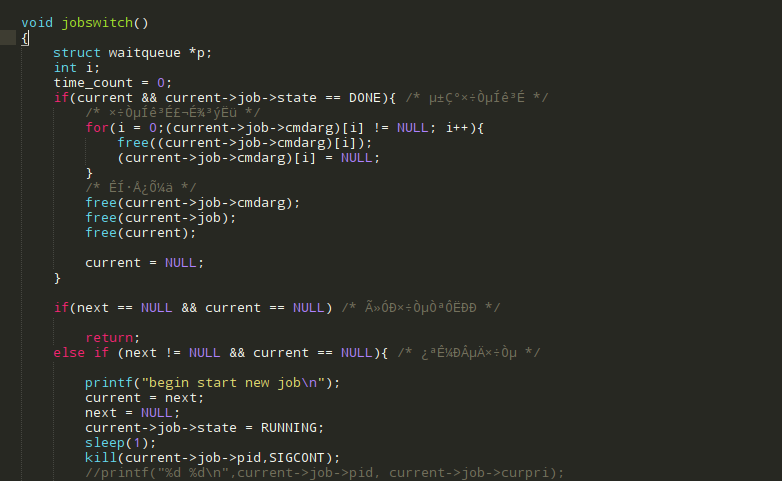


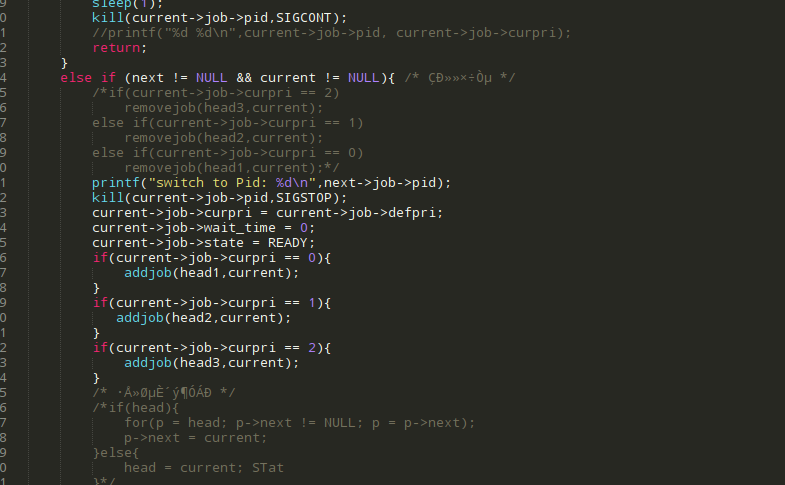
对jobselect的修改。

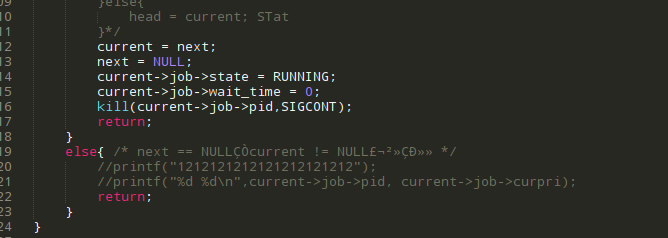




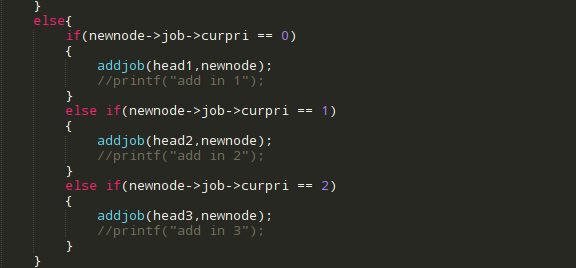
对jobswitch的修改。



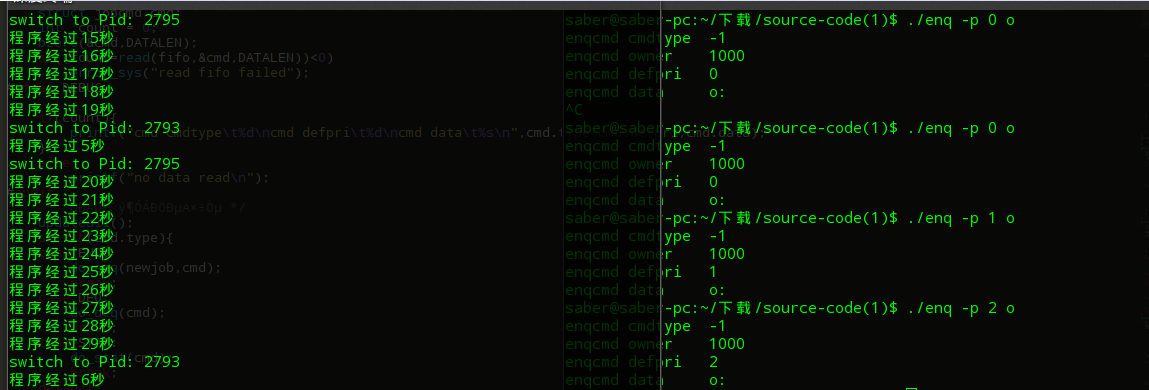




对于enq函数的修改。

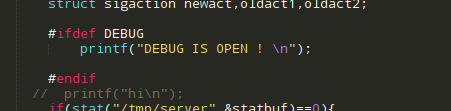


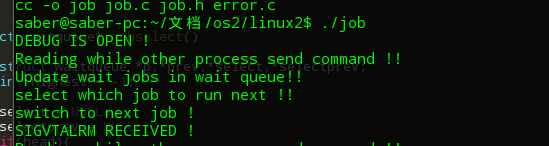
程序运行截图。



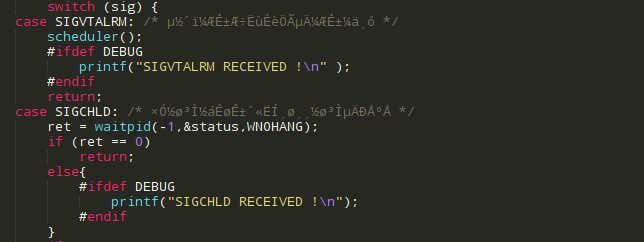
2. 实现调试任务：

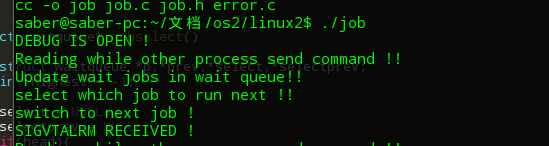
（1）代码及运行截图



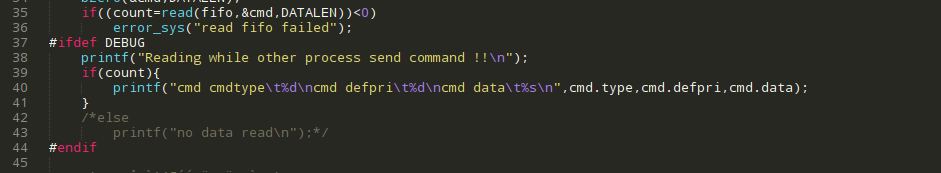


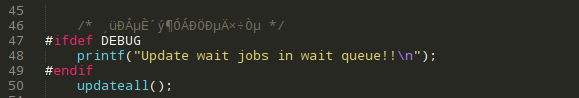
（2）代码及运行截图

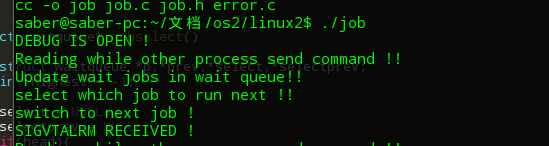


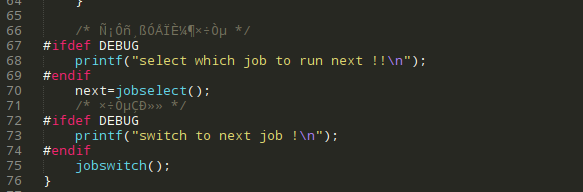


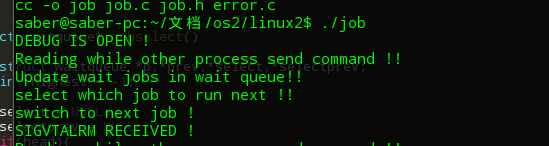
（3）代码及运行结果



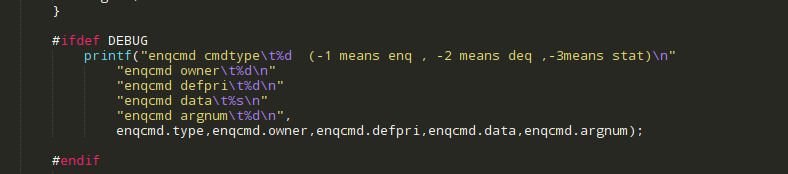


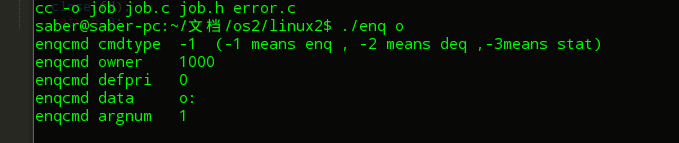






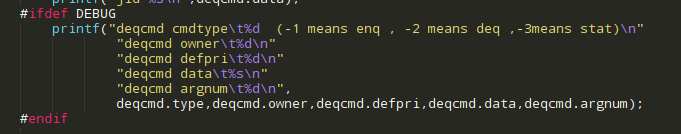
（4）代码及运行结果

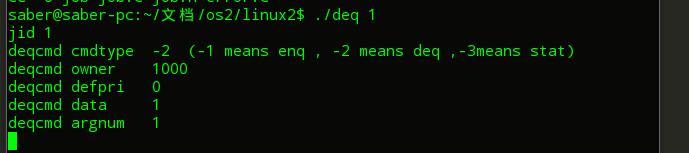




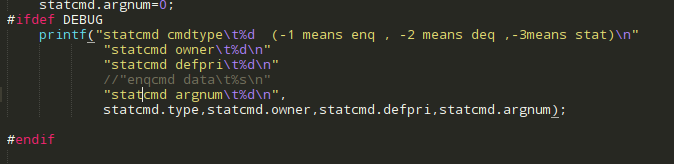
（5）代码及运行结果

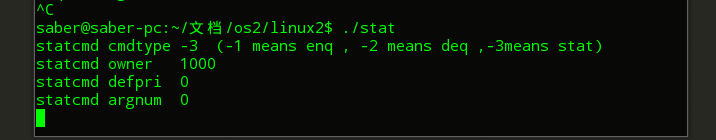
Deq





Stat

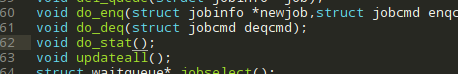




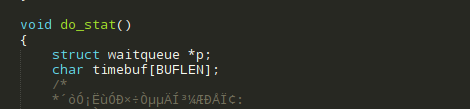
（6）代码及运行及如果

修改了do\_stat函数用这个函数来方便输出

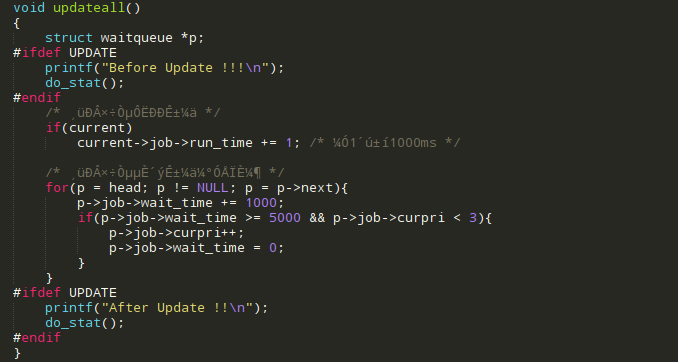
修改job.h



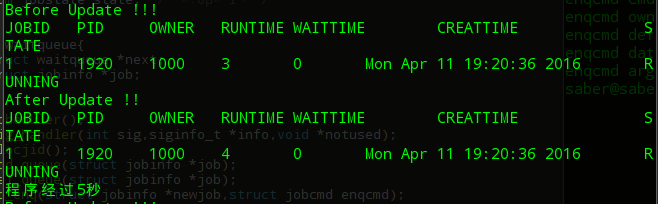
修改do\_stat



代码

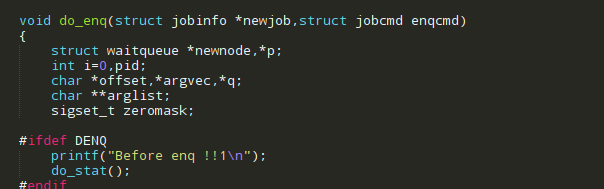


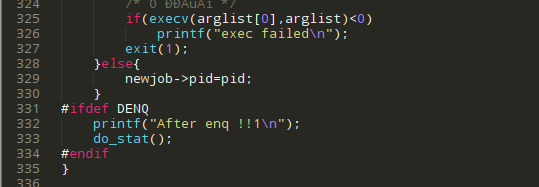
结果：

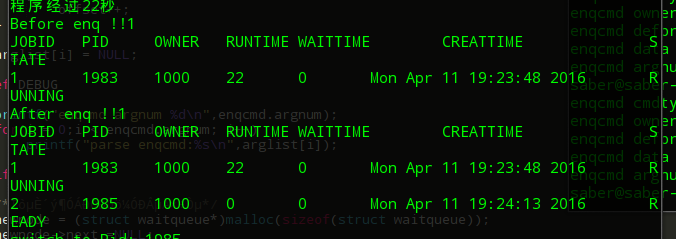


（7）代码及运行结果

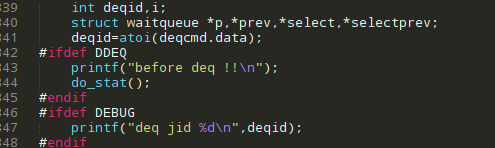
Enq

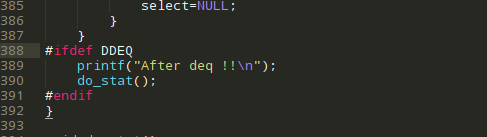






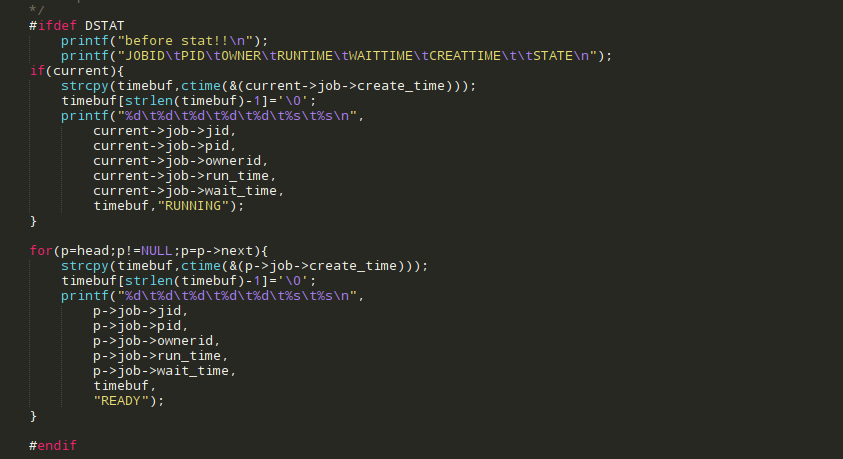
Deq



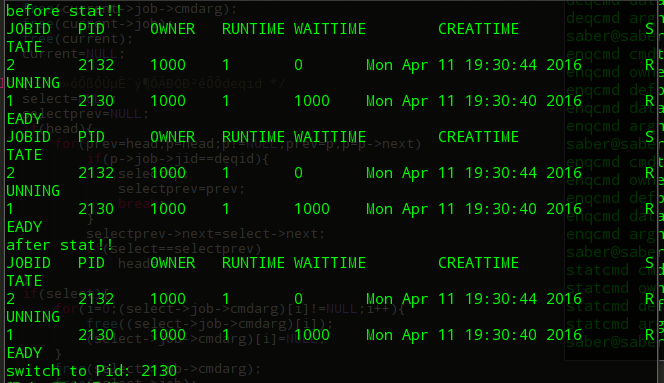




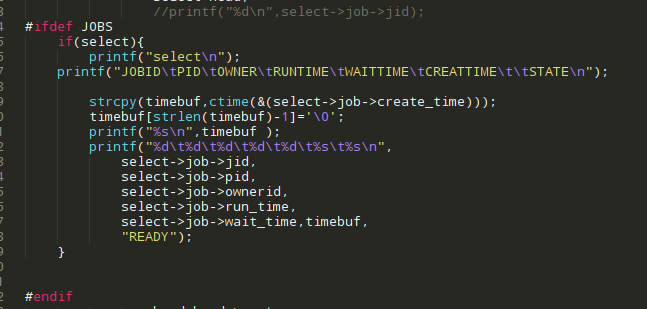
Stat

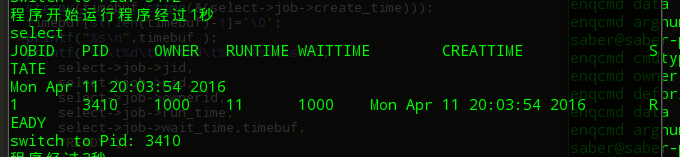




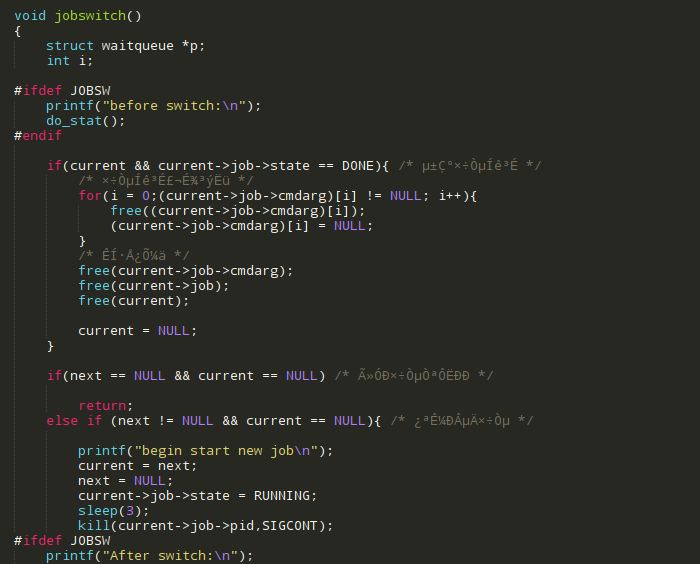


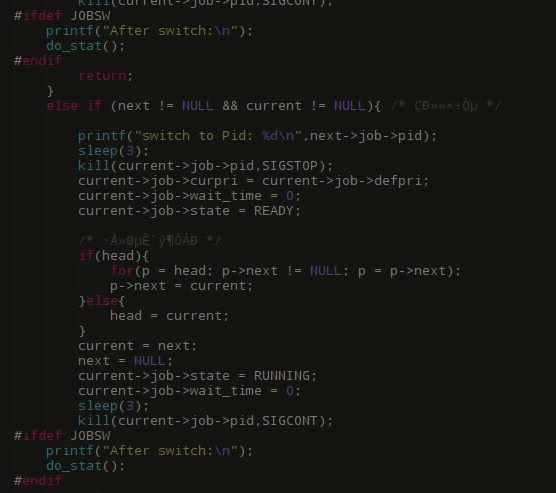
（8）代码及运行结果

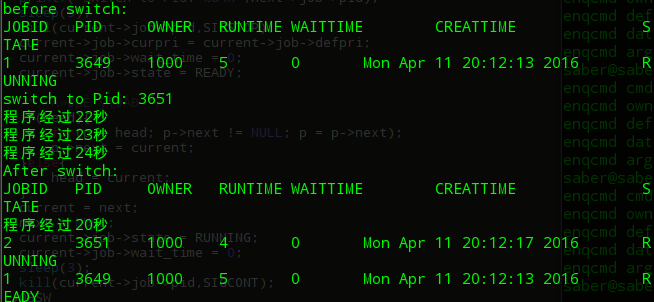




（9）代码及运行结果

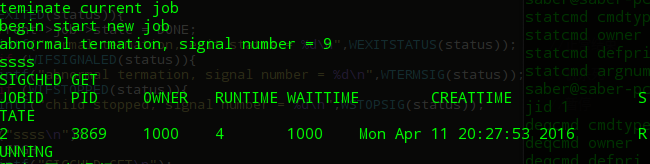






（10）代码及运行后的结果



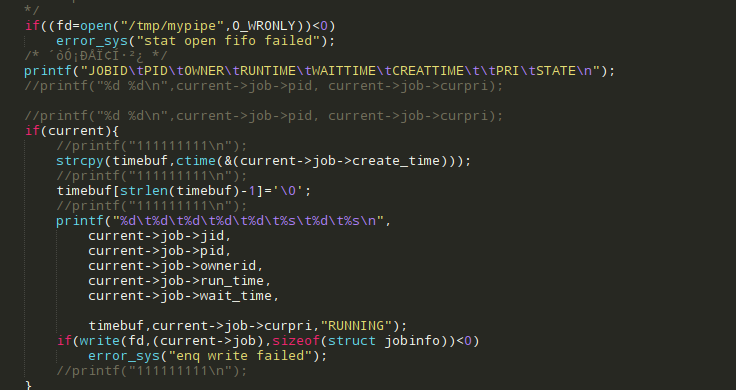


3.实现stat的管道通信。

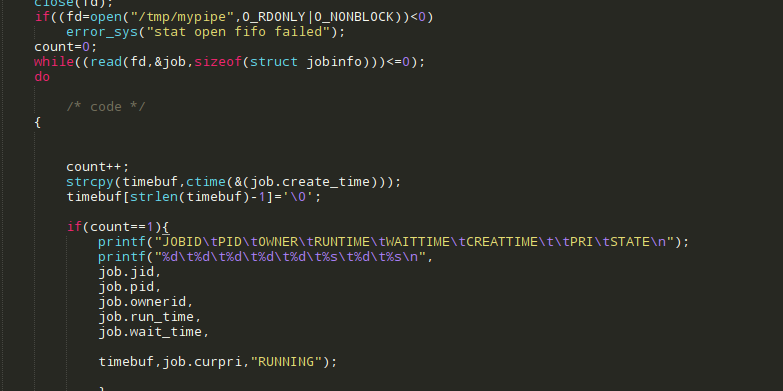
在main函数中新建一个管道

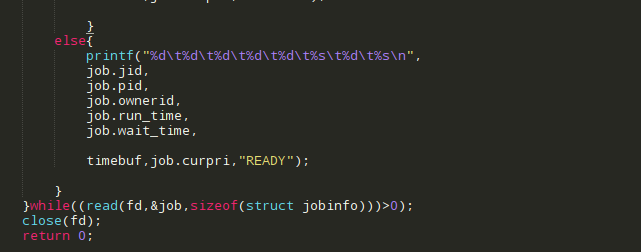


当stat调用时写入管道

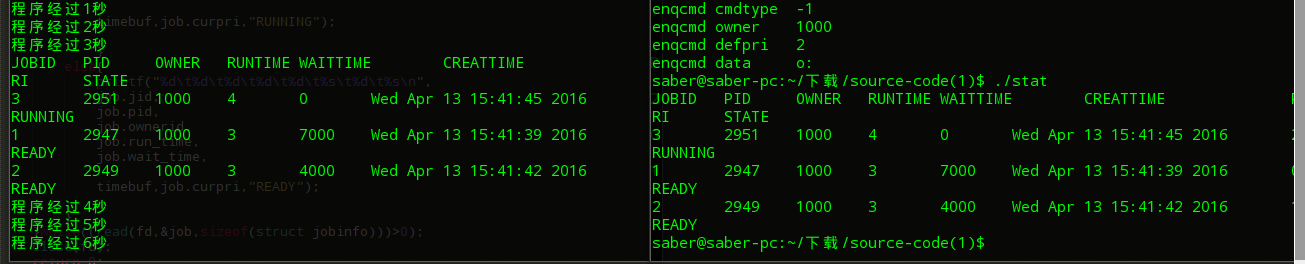


在stst.c接受管道信息输出。





程序运行结果：



# 4.收获和感想

经过这次实验，我们了解到了cpu对于作业的的调度，通过实现多级轮转的算法，我们加深了cpu在处理多个作业的时候的调度机制。

在这次作业中，我们还加深了对于管道的认识。更加了解了管道在系统信息传递中的作用。

而且我们还学到了在linux系统中编程的调试方法，而不是再像以前一样导出打印，这让我们以后的调试工作变得更加简单。

在这次试验中，我们小组成员协调配合。将任务分解，然后组合，经验互相分享，最后我们在加深了对于linux系统的了解。不仅学到了我嗯这次实验本应该学到的内容，也体会到了合作学习的乐趣。