西安交通大学

操作系统专题实验报告

班级： 计算机2106

学号： 2213712884

姓名： 张昌宇

2023年 12 月 10 日

目 录

1 openEuler 系统环境实验.................................................................................................................1

1.1 实验目的 ............................................................................................................................... 1

1.2 实验内容 ............................................................................................................................... 1

1.3 实验思想 ............................................................................................................................... 2

1.4 实验步骤 ............................................................................................................................... 2

1.5 测试数据设计....................................... .................................................................................4

1.6 程序运行初值及运行结果分析.............................................................................................9

1.7 实验总结 ............................................................................................................................. 13

1.7.1 实验中的问题与解决过程...................................................................................... 13

1.7.2 实验收获.................................................................................................................. 14

1.7.3 意见与建议.............................................................................................................. 14

1.8 附件 ..................................................................................................................................... 14

1.8.1 附件 1 程序............................................................................................................. 14

1.8.2 附件 2 Readme......................................................................................................... 30

2 进程通信与内存管理..................................................................................................................... 44

2.1 实验目的 ............................................................................................................................. 44

2.2 实验内容 ............................................................................................................................. 45

2.3 实验思想 ............................................................................................................................. 46

2.4 实验步骤 ............................................................................................................................. 47

2.5 测试数据设计.......................................................................................................................48.

2.6 程序运行初值及运行结果分析...........................................................................................50.

2.7 页面置换算法复杂度分析...................................................................................................56

2.8 回答问题 ............................................................................................................................. 57

2.8.1 软中断通信.............................................................................................................. 57

2.8.2 管道通信.................................................................................................................. 58

2.9 实验总结 ............................................................................................................................. 59

2.9.1 实验中的问题与解决过程...................................................................................... 59

2.9.2 实验收获.................................................................................................................. 60

2.9.3 意见与建议.............................................................................................................. 60

2.10 附件 ................................................................................................................................... 61

2.10.1 附件 1 程序........................................................................................................... 61

2.10.2 附件 2 Readme....................................................................................................... 84

3 文件系统 ........................................................................................................................................ 98

3.1 实验目的 ............................................................................................................................. 98

3.2 实验内容 ............................................................................................................................. 98

3.3 实验思想 ............................................................................................................................. 98

3.4 实验步骤 ........................................................................................................................... 103

3.5 程序运行初值及运行结果分析.........................................................................................105.

3.6 实验总结 ........................................................................................................................... 110

3.6.1 实验中的问题与解决过程.................................................................................... 110

3.6.2 实验收获................................................................................................................ 110

I

3.6.3 意见与建议............................................................................................................ 110

3.7 附件 ................................................................................................................................... 111

3.7.1 附件 1 程序........................................................................................................... 111

3.7.2 附件 2 Readme....................................................................................................... 150

**1 openEuler 系统环境实验**

**1.1 实验目的**

（1）熟悉基于鲲鹏架构弹性云服务器 ECS 上 openEuler 操作系统基本系统环境；

（2）理解进程的创建、进程地址空间的概念、父子进程资源的关系；

（3）观察进程调度，了解进程调度的过程，了解孤儿进程和僵尸进程的区别；

（4）理解线程与进程的关系，对等线程间的关系。

（5）理解并运用信号量及其 PV 操作、自旋锁实现线程间的同步互斥。

**1.2 实验内容**

（1）在华为云和本地虚拟机上搭建实验环境

（2）进程相关编程实验

①熟悉操作命令、编辑、编译、运行程序。完成教材中所给代码程序的运行 验证， 多运行几次程序观察结果;去除 wait 后再观察结果并进行理论分析。

②(a) 添加一个全局变量并在父进程和子进程中对这个变量做不同操作，输出操作结果并解释;

(b) 在 return 前增加对全局变量的操作并输出结果，观察并解释;

(c) 修改程序体会在子进程中调用 system 函数和在子进程中调用 exec 族函数;

（3）线程相关编程实验   
①在进程中给一变量赋初值并成功创建两个线程;

②在两个线程中分别对此变量循环五千次以上做不同的操作(自行设计) 并输出结果;

③多运行几遍程序观察运行结果，如果发现每次运行结果不同，请解释原 因并修改程序解决， 考虑如何控制互斥和同步;

④将任务一中第一个实验调用 system 函数和调用 exec 族函数改成在线 程中实现，观察运 行结果输出进程 PID 与线程 TID 进行比较并说明原因。

**1.3 实验思想**

openEuler 基于 Linux 内核，支持多种架构，支持多种虚拟化技术，支持多种容器技术，支 持多种云计算技术。openEuler 采用 Linux 内核，通过了解 openEuler 系统下的 shell 命令和系 统命令后，所学习的知识可以轻松迁移到其他 Linux 发行版本中，如 Ubuntu。

通过配置基于鲲鹏架构弹性云服务器 ECS 上 openEuler 操作系统基本系统环境，运行 shell 命令查看系统信息以达到了解和使用 openEuler 操作系统的目的。 通过进程相关编程实验，编写和运行简单的进程、线程相关程序，理解进程与线程概念、 进程空间与物理内存空间、进程调度、进程间变量管理、进程调用其他程序、如何实现正确的 并发（同步互斥）等方面在操作系统中的实际操作。具体来讲，通过输出子进程和父进程的 PID， 观察进程调度，了解进程调度的过程，了解孤儿进程和僵尸进程的区别。观察并发进程中的全 局变量改变，输出父子进程共享变量地址以了解物理地址与虚地址概念，从而理解进程地址空 间的概念，了解关于地址绑定的基础知识。创建两个线程运行后体会线程与进程的关系、对等 线程间的资源共享以及同步与互斥的知识。

**1.4 实验步骤**

（1）登录华为云,搭建 openEuler 操作系统环境。

（2）编辑、编译、运行给定的程序，多次运行观察实验结果。去除 wait()后观察结果并进行理论分析, 比较程序在有无 wait()函数时的运行结果，分析 wait()函数的作用。

（3）添加一个全局变量，在父进程和子进程中对这个变量做不同操作，输出操作结果，同时输 出两种变量的地址并进行分析。

（4）在子进程中调用 system 函数执行自己写的一段程序，在此程序中输出进程 PID 和其父进 程 PID 进行比较分析。

（5）在子进程中调用 exec 函数执行自己写的一段程序，在此程序中输出进程 PID 和其父进程 PID 进行比较分析。

（6）修改给定程序，给一变量赋初值并创建两个线程，在两个线程中分别对此变量循环 5000 次做不同的操作并输出结果。

（7）多运行几遍程序观察运行结果。

（8）完成上述过程的同步和互斥操作。

（9）在两个线程中调用 system 函数执行自己写的一段程序，在此程序中输出线程 TID 及进程 PID,进行分析。

（10）在两个线程中调用 exec 函数执行自己写的一段程序，在此程序中输出线程 TID 及进程 PID,进行分析。

（11）参考实验指导书，编写模拟自旋锁程序代码，补充主函数代码，用自旋锁实现线程间 的同步。编译并运行程序，分析运行结果

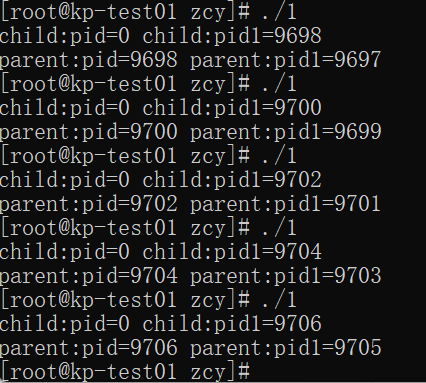
**1.5 测试数据设计**

测试数据都是随机选取，为了体现进程线程的运行次序以及中断的次序；并且选择数 据的时候主要选择整数以方便计算。

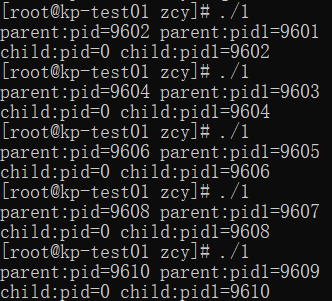
**1.6 程序运行初值及运行结果分析**

**1.6.1 进程相关编程实验**

1.编写并多次运行图 1-1 中代码

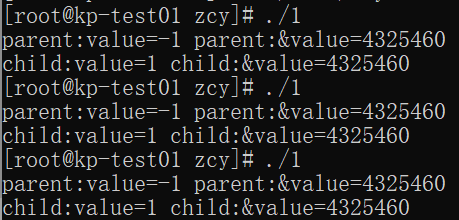


2.删去wait( )多次运行



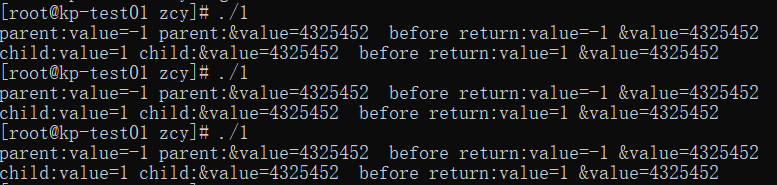
**分析：由运行结果分析可知，当保留wait(NULL)时，父进程会等待子进程程序运行结束再进行，所以是子进程先结束，然后父进程才会结束；而当删去wait(NULL)之后，父进程不必等待子进程，因此父进程先结束，子进程后结束。**

3.修改代码，增加一个全局变量并在父子进程中对其进行不同的操作



**分析：fork时，子进程拷贝父进程的数据空间中value值，此时父子进程中value都为0；然后父进程中value--得到-1，子进程中value++得到1，这说明父子进程数据空间的内容是相互独立的；同时可以发现父子进程中value的虚拟地址是相同的，这是为了保证子进程中依旧可以使用原程序中的指针、函数、寻址变量等数据或功能。**

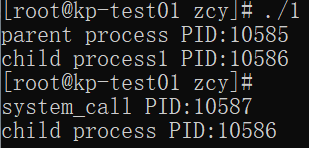
4.在 return 前增加对全局变量的操作并输出结果



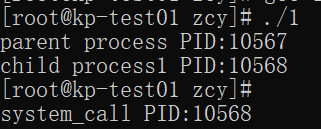
**解释：由结果可知，fork()之后父子进程直至结束前数据空间都是独立的，而其虚拟地址是相同的。**

5.在子进程中调用 system()与 exec 族函数。编写system\_call.c 文件输出进程号 PID，编译后生成 system\_call 可执行文件。在子进程中调用 system\_call,观察输出结果并分析总结。

调用system()



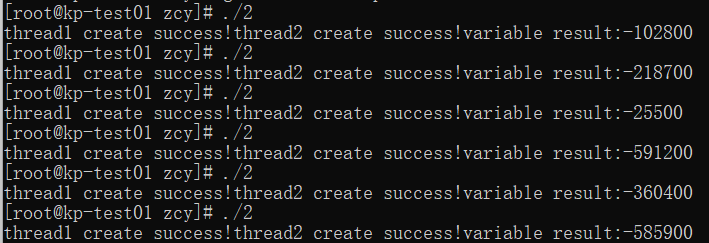
调用execl()



**解释：由结果可知及相关知识可知，system()函数运行参数命令时会创建一个新的子进程这个子进程有自己专门的pid，函数创建的子进程结束后会继续执行原来的进程；而execl()函数不会创建新的子进程，而是将该进程完全替换为新进程，因此既不会产生新的pid，也不会在execl()执行完后继续执行原来的进程。**

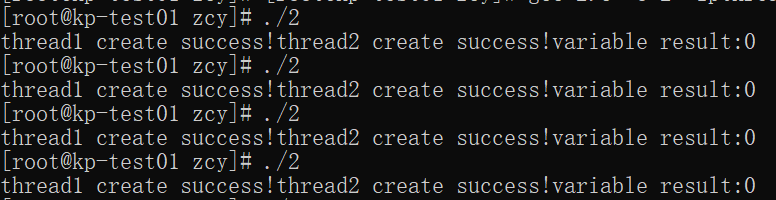
**1.6.2 线程相关编程实验**

1.设计程序，创建两个子线程，两线程分别对同一个共享变量多次操 作，观察输出结果



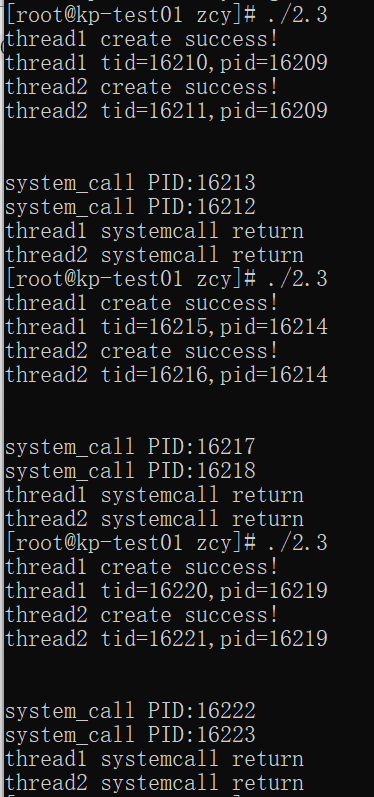
**分析：在同一个进程中的两个线程并发执行，并且同时调用共享资源value并对其数值进行操作，就会导致竞争现象的出现，不同线程的操作交叉执行，导致结果不稳定，每次运行会得到不同的结果。**

2.修改程序，定义信号量 signal，使用 PV 操作实现共享变量的访问 与互斥。运行程序，观察最终共享变量的值。



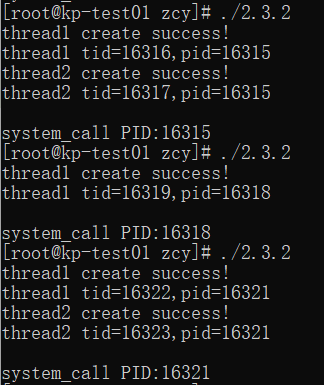
3.在第一部分实验了解了 system()与 exec 族函数的基础上，将这两个函数的调用改为在线程中实现，输出进程 PID 和线程的 TID 进行分析。

system()



**分析：同一个进程中每个线程都占有独立的TID，不同线程之间的TID不同；而同一进程中的两个不同线程输出进程号PID时，因为他们在同一个进程中所以PID是相同的。**

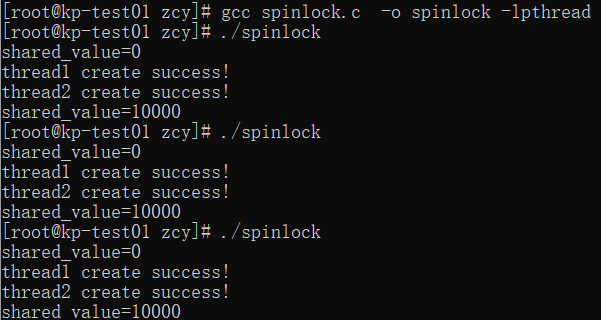
execl()



**对比分析：system()函数运行时会创建一个新的子进程这个子进程有自己专门的pid，函数创建的子进程结束后会继续执行原来的进程；而execl()函数不会创建新的子进程，而是将该进程完全替换为新进程，因此既不会产生新的pid，也不会在execl()执行完后继续执行原来的进程。**

**1.6.3 自旋锁实验**

1.根据实验内容要求，编写模拟自旋锁程序代码 spinlock.c

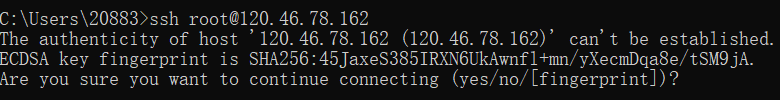


**分析：由于自旋锁的存在，当某个线程操作共享变量的时候会给共享变量加锁，此时其他线程想要操作共享变量时会进行自旋等待，由此避免了多个线程同时操作造成竞争现象。**

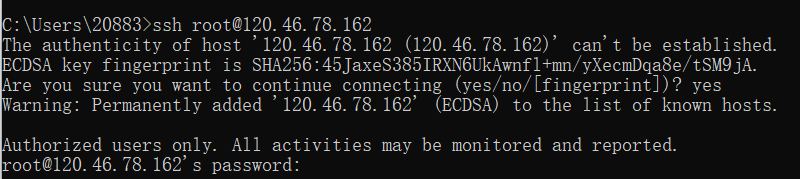
**1.7 实验总结**

**1.7.1 实验中的问题与解决过程**

**问题一**：使用ssh远程连接服务器失败

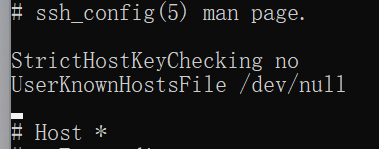


初步解决：输入yes，然后回车即可

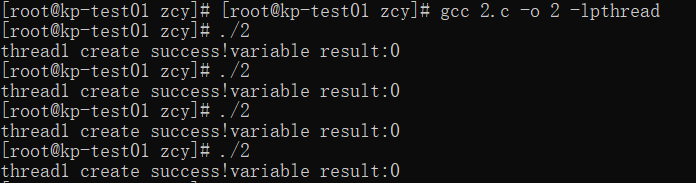


最终解决：修改配置文件





**问题二**：1.6.2.1中每次输出都为0，并且线程2并没有正常创建，程序也不报错



解决：经过调试后发现是因为主线程结束过快,没有等待线程结束。后查阅资料后加上pthread\_join函数即可

**1.7.2 实验收获**

通过本次实验我了解了进程与线程。熟悉了进程和线程的 创建、子进程与父进程在运行时的关系、僵尸进程、孤儿进程等知识，也验证了虚拟地址与物理地址的区别，认识了系统调用。还学习了自旋锁的相关知识，极大的激发了我对于接下来实验的兴趣。

**1.7.3 意见与建议**

暂无

**1.8 附件**

**1.8.1 程序代码**

由于使用虚拟机，源程序全部粘贴在readme中

链接：https://pan.baidu.com/s/1x2gdJ3dzYI-GuMpFrc3BMw?pwd=d9tb 提取码：d9tb

**1.8.2 readme**

链接：https://pan.baidu.com/s/1x2gdJ3dzYI-GuMpFrc3BMw?pwd=d9tb 提取码：d9tb

**2 进程通信与内存管理**

**2.1 实验目的**

（1）.进程的软中断通信： 编程实现进程的创建和软中断通信，通过观察、分析实验现象，深入理解进程及进程在调 度执行和内存空间等方面的特点，掌握在 POSIX 规范中系统调用的功能和使用。

（2）进程的管道通信： 编程实现进程的管道通信，通过观察、分析实验现象，深入理解进程管道通信的特点，掌 握管道通信的同步和互斥机制。

（3）内存管理： 1) 体会操作系统中内存的分配和回收模式； 2) 理解并掌握内存分配的三种基本算法（首次适应（FF），最佳适应（BF）,最差适应 （WF））； 3) 学会进程的建立，当一个进程被终止时内存是如何处理被释放块，并当内存不满足进程申 请时是如何使用内存紧凑； 4) 掌握内存回收过程及实现方法； 5) 学会进行内存的申请释放和管理；

**2.2 实验内容**

1.进程的软中断通信

（1）使用 man 命令查看 fork 、kill 、signal、sleep、exit 系统调用的帮助手 册。

（2）根据流程图编制实现软中断通信的程序：使用系统调用 fork()创建 两个子进 程，再用系统调用 signal()让父进程捕捉键盘上发出的中断信号（即 5s 内按下 delete 键或 quit 键），当父进程接收到这两个软中断的某一个后，父进程用系统 调用 kill() 向两个子进程分别发出整数值为 16 和 17 软中断信号，子进程获得 对应软中断信号，然后分 别输出下列信息后终止： Child process 1 is killed by parent !! Child process 2 is killed by parent !! 父进程调用 wait()函数 等待两个子进程终止后，输出以下信息，结束进程执行： Parent process is killed!! 注： delete 会向进程发送 SIGINT 信号，quit 会向进程发送 SIGQUIT 信号。 ctrl+c 为 delete，ctrl+\为 quit 。

（3）多次运行所写程序，比较 5s 内按下 Ctrl+\或 Ctrl+Delete 发送中断，或 5s 内不进行 任何操作发送中断，分别会出现什么结果？分析原因。

（4）将本实验中通信产生的中断通过 14 号信号值进行闹钟中断，体会不同中断的 执行样 式，从而对软中断机制有一个更好的理解。

2. 进程的管道通信

（1）学习 man 命令的用法，通过它查看管道创建、同步互斥系统调用的在线帮 助，并阅读 参考资料。

（2）根据流程图和所给管道通信程序，按照注释里的要求把代码补充完 整，运行 程序，体会互斥锁的作用，比较有锁和无锁程序的运行结果，分析管道通信是如何 实现同步与互斥的。

3. 内存管理

（1）理解内存分配 FF，BF，WF 策略及实现的思路。

（2）参考给出的代码思路，定义相应的数据结构，实现上述 3 种算法。每种算法 要实现内 存分配、回收、空闲块排序以及合并、紧缩等功能。

（3）充分模拟三种算法的实现过程，并通过对比，分析三种算法的优劣。

**2.3 实验思想**

软中断通信通过系统调用 signal()让父进程捕捉键盘上发出的中断信号，让软中断函数与输出 函数相关联，之后进行信号的收发。

管道通信先设置一个无名管道，然后在无名管道中进行数字或者字符的写入与读出。

内存管理根据实验内容的要求，一一实现出合并、紧缩等功能；此外根据三种算法，模拟出 其实现过程。

**2.4 实验步骤**

1. 软中断通信

（根据指导书编写软中断的程序，分别向其发送 SIGINT 和 SIGQUIT 信号，此外再不对其发送 信号，比较三种情况下的不同结果并分析。 将本实验中通信产生的中断通过 14 号信号值进行闹钟中断，体会不同中断的执行样式。

2. 管道通信

先猜想一下这个程序的运行结果。分析管道通信是怎样实现同步与互斥的，然后按照注释里的要求把代码补充完整，运行程序。修改程序并运行，体会互斥锁的作用，比较有锁和无锁程序的运行结果，并解释之。

3. 内存管理

首先对三种算法进行深度的理解，根据实验指导书补全代码并运行分析结果。

**2.5 测试数据设计**

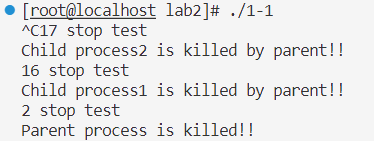
测试数据中，根据实验要求，在软中断通信的实验中的循环次数数据选择的是大于要求值 的任意值；在管道通信中输入的数据为任意的字符数据

**2.6 程序运行初值及运行结果分析**

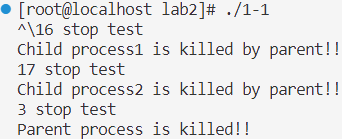
**2.6.1 进程的软中断通信**

1.根据流程图编制实现软中断通信的程序

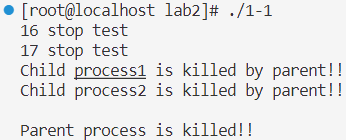
运行结果：  
手动crtl+c



手动crtl+\

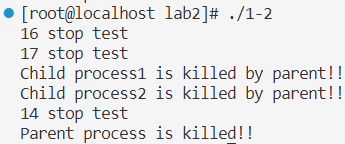


等待5s自动停止



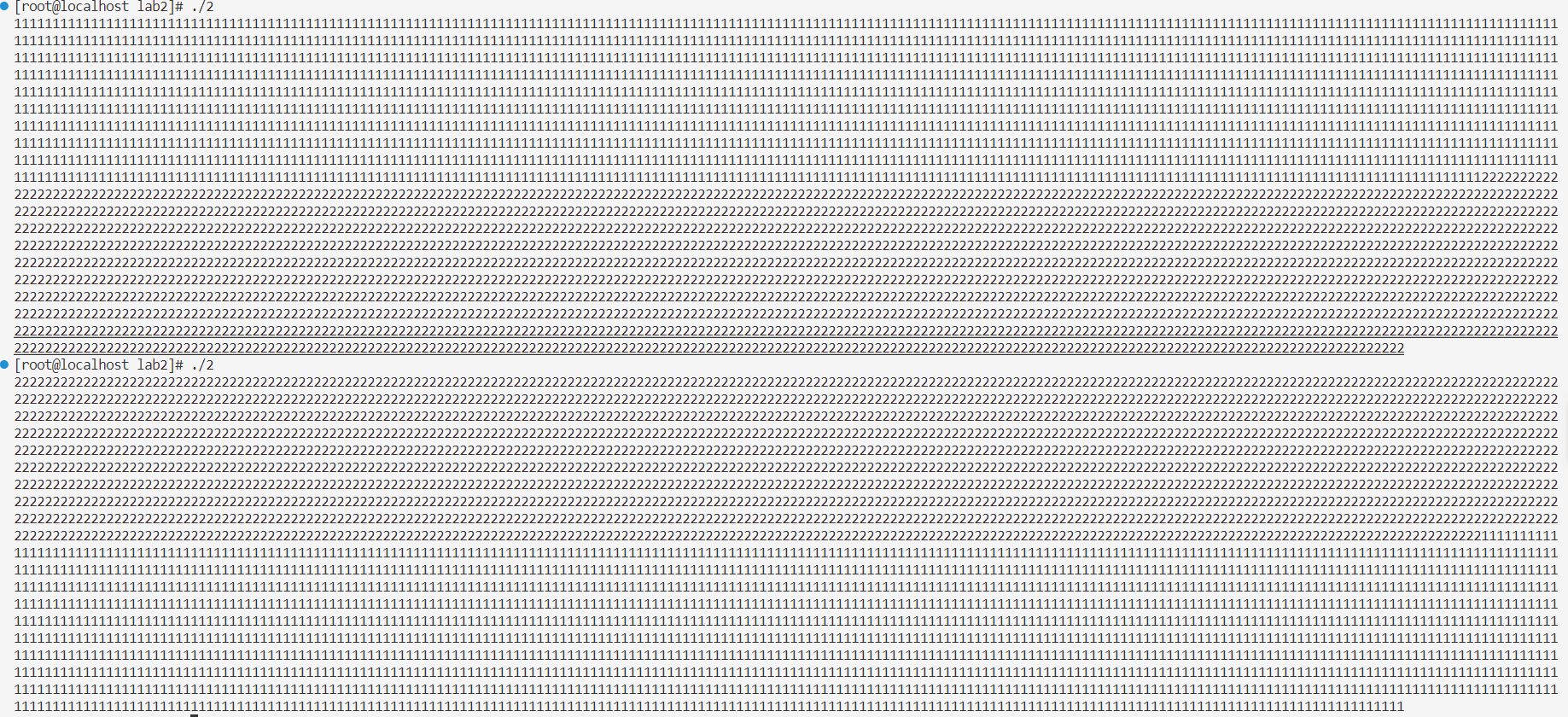
**分析：5s内手动crtl+c时，会向中断挂起状态，然后向两个子进程分别发送16和17信号，两个子进程接收对应信号后，flag变为0，中断等待状态，继续运行至结束，父进程待到两个子进程都结束后，也运行结束；如果等待5s，直至sleep(5)所导致的挂起状态结束，父进程就会自动按顺序向下运行，后续过程与手动结束相同。**

2.通过 14 号信号值进行闹钟中断

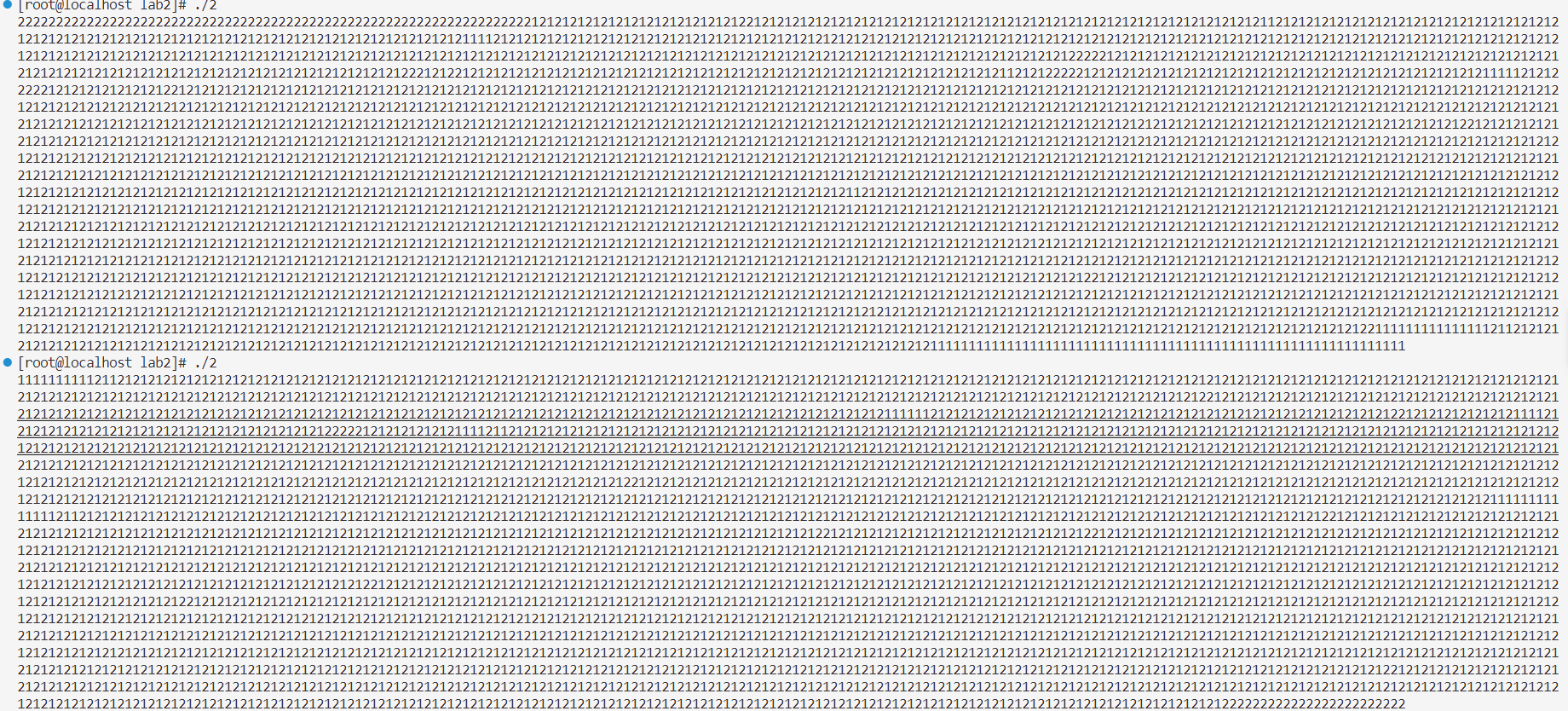
****

**分析：父进程设置了一个5s的计时器，当计时器结束时会向父进程发送一个闹钟中断信号SIGALRM，父进程捕捉到这个信号后执行相关函数，继续向下执行，后续过程与2.6.1-1相同。2.6.2 进程的管道通信**

1.有锁



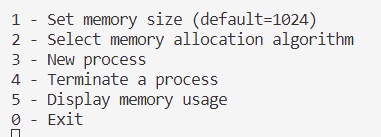
2.无锁



**分析：由结果来看，有锁时会随机选择写入2000个‘1’或者2000个‘2’，这个选择是随机的，但是一旦开始写入就会等2000个‘1’或者2000个‘2’全部写入之后才会写入另外一个，这是因为子进程在写入之前给管道加锁，等写入完成后才给管道解锁，加锁期间另一个进程无法进行写入；而无锁的时候下一个写入的字符完全是随机的，有可能是‘1’也可能是‘2’，因为不加锁管道对两个进程同时都是开放的，在某一个进程通过管道写入的时候，此时如果另一个进程要写入，就会进入等待队列，一旦管道空闲就会立即写入，所以最终结果是随机的。**

**2.6.3 内存的分配与回收**

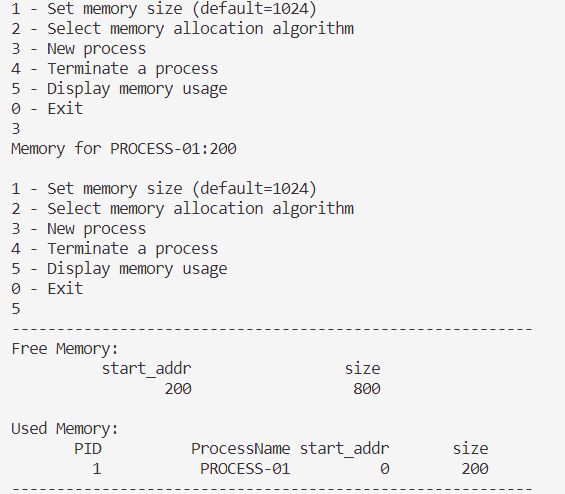
初始界面



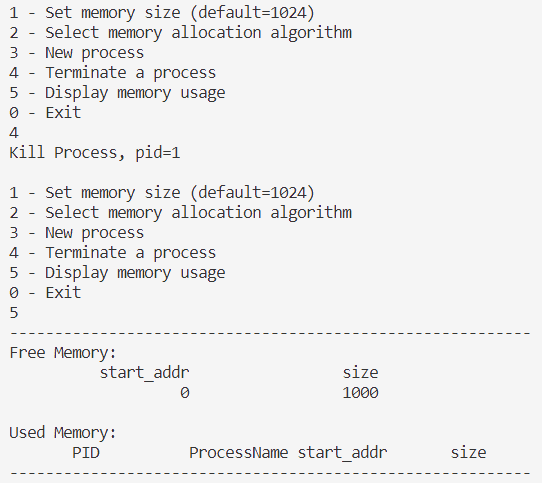
设置内存大小



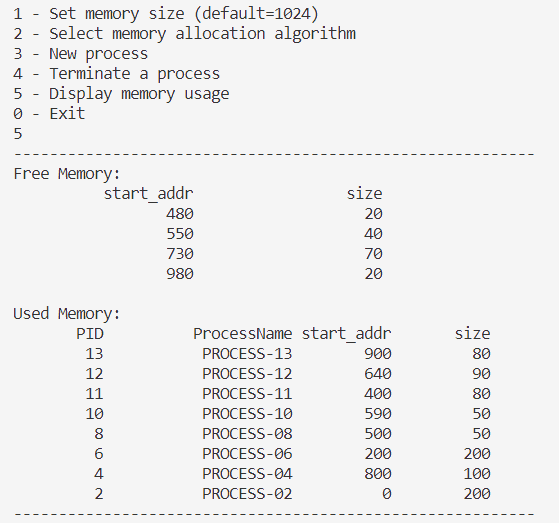
创建进程



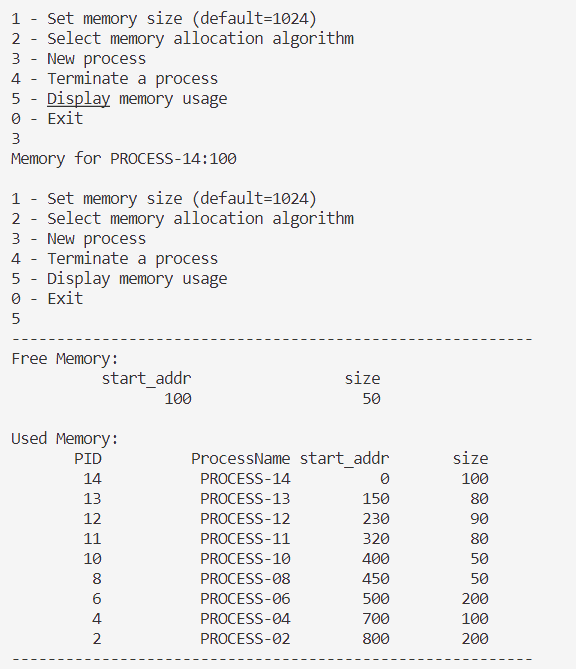
删除进程



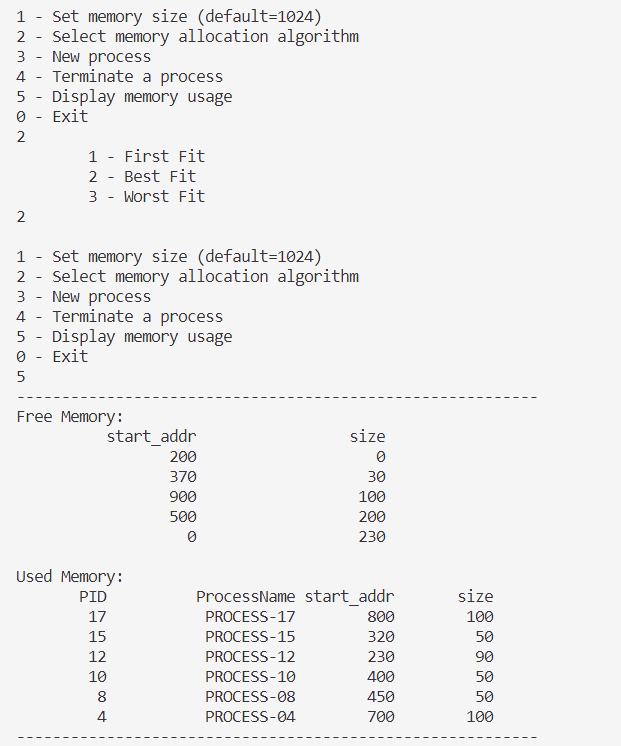
内存展示



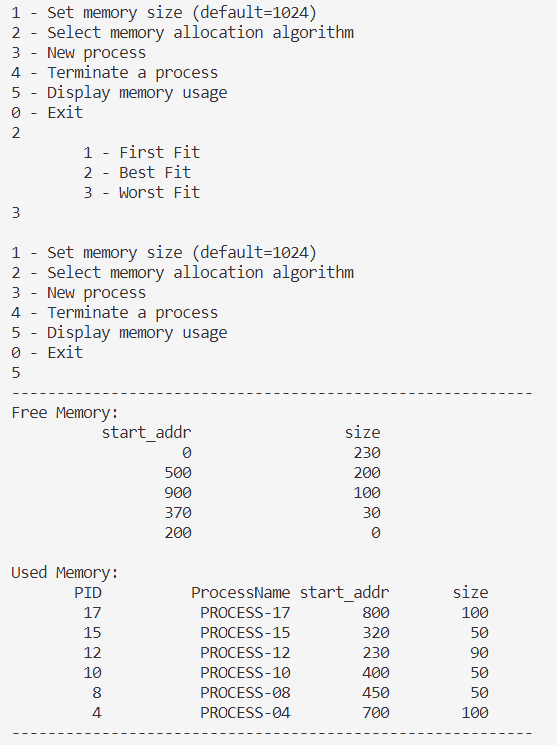
内存紧缩



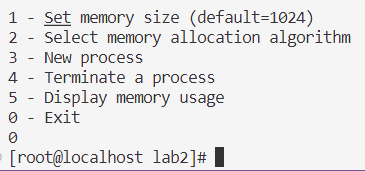
BF



WF



退出系统



**2.7 回答问题**

1.软中断通信

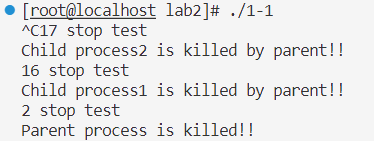
（1）你最初认为运行结果会怎么样？写出你猜测的结果。

子进程1、2先结束，父进程再结束

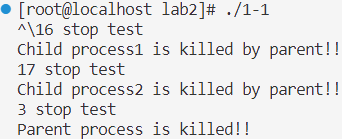
（2）实际的结果什么样？有什么特点？在接收不同中断前后有什么差别？请将 5 秒内中断 和 5 秒后中断的运行结果截图，试对产生该现象的原因进行分析。

5秒内

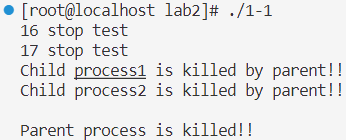
手动crtl+c



手动crtl+\

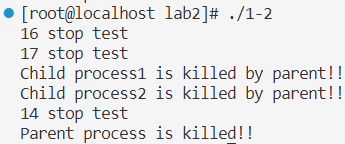


等待5秒后



分析：5s内手动crtl+c时，会向中断挂起状态，然后向两个子进程分别发送16和17信号，两个子进程接收对应信号后，flag变为0，中断等待状态，继续运行至结束，父进程待到两个子进程都结束后，也运行结束；如果等待5s，直至sleep(5)所导致的挂起状态结束，父进程就会自动按顺序向下运行，后续过程与手动结束相同。

（3）改为闹钟中断后，程序运行的结果是什么样子？与之前有什么不同？

****

父进程的结束方式不同，时钟中断信号为14

（4）kill 命令在程序中使用了几次？每次的作用是什么？执行后的现象是什么？

2次．分别给子进程1和子进程2传递软中断信号。执行后两个子进程接收到软中断信号进行一系列操作结束挂起，表现为被父进程杀死

（5）使用 kill 命令可以在进程的外部杀死进程。进程怎样能主动退出？这两种退出方式 哪种更好一些？

使用exit();kill更好一些，因为可以通过中断信号进行控制

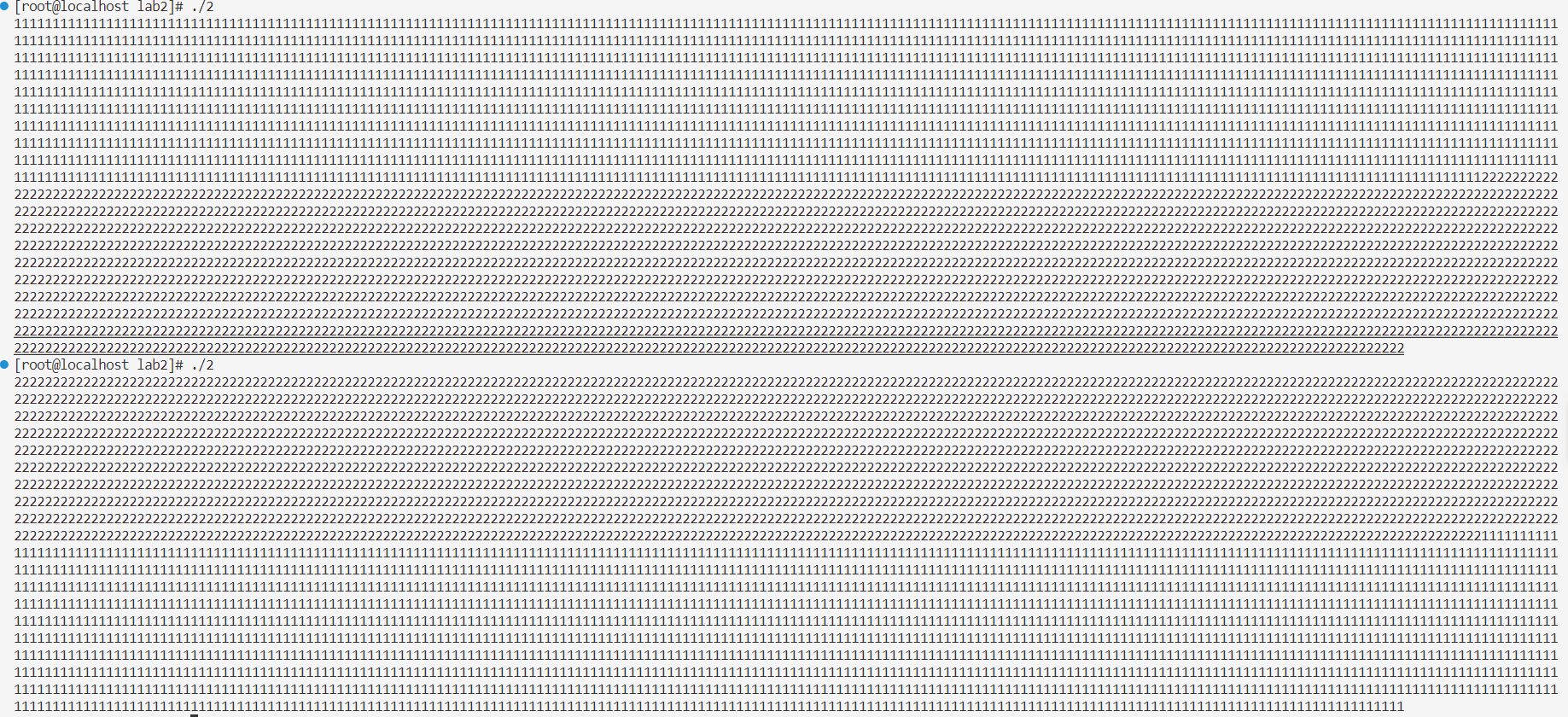
2.管道通信：

(1) 你最初认为运行结果会怎么样？

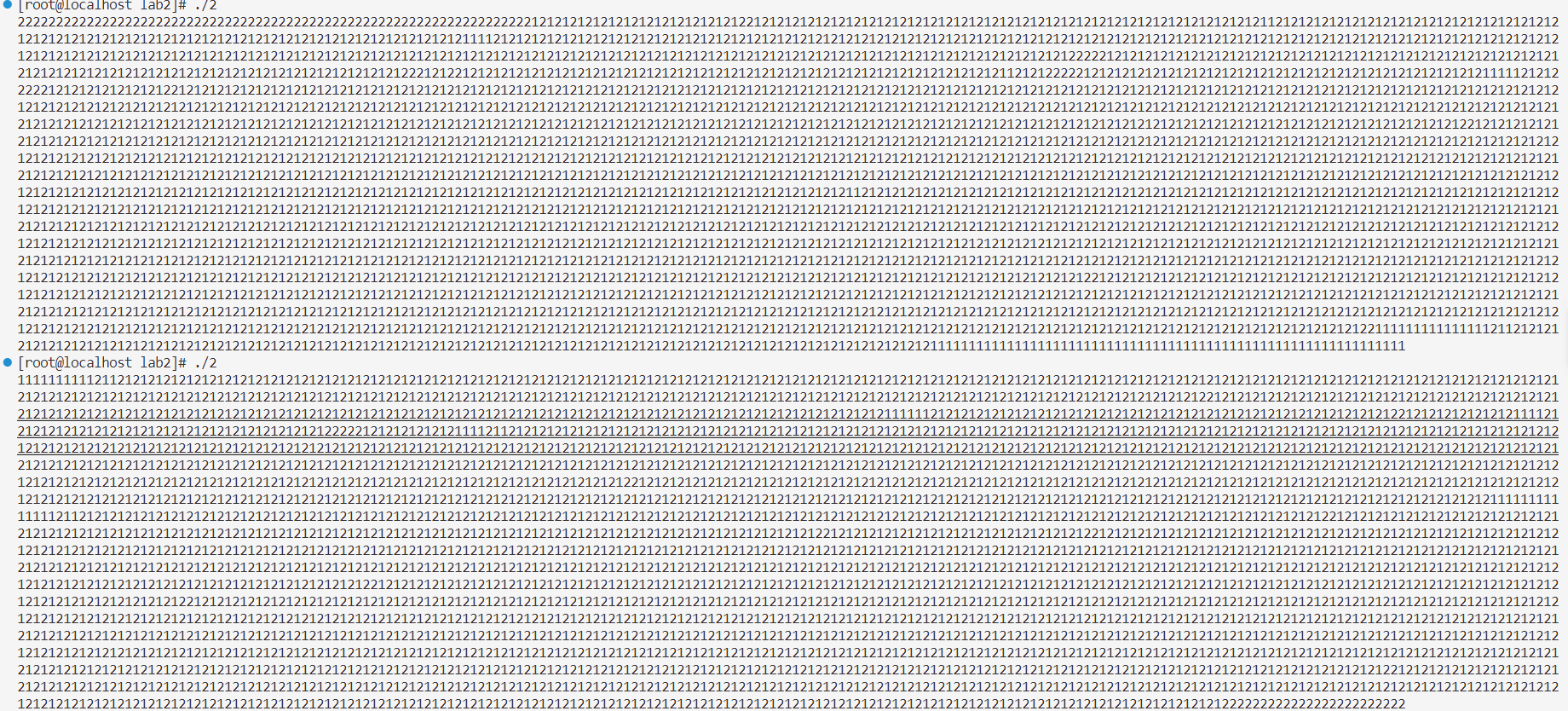
输出1000个1和1000个2

(2) 实际的结果什么样？有什么特点？试对产生该现象的原因进行分析。

有锁



无锁



分析：由结果来看，有锁时会随机选择写入2000个‘1’或者2000个‘2’，这个选择是随机的，但是一旦开始写入就会等2000个‘1’或者2000个‘2’全部写入之后才会写入另外一个，这是因为子进程在写入之前给管道加锁，等写入完成后才给管道解锁，加锁期间另一个进程无法进行写入；而无锁的时候下一个写入的字符完全是随机的，有可能是‘1’也可能是‘2’，因为不加锁管道对两个进程同时都是开放的，在某一个进程通过管道写入的时候，此时如果另一个进程要写入，就会进入等待队列，一旦管道空闲就会立即写入，所以最终结果是随机的。

(3) 实验中管道通信是怎样实现同步与互斥的？如果不控制同步与互斥会发生什 么后果？

利用互斥锁。如果不控制，不同进程写入的信息会交替出现，导致不同写入的数据都无法分辨和使用。

**2.8 实验总结**

**2.8.1 实验遇到的问题与解决**

**问题**：不输出子进程结束标志，只有父进程结束



解决：没有在两个子进程中忽略crtl+c和crtl+z信号，导致子进程直接结束，没有输出；只需要在两个子进程中添加signal(SIGINT,SIG\_IGN);signal(SIGQUIT,SIG\_IGN) 来将两个中止信号忽略掉。

**2.8.2 实验收获**

学习了软中断通信和管道通信等进程间的通讯方式，还理解并模拟了内存分配与管理等内容，受益匪浅

**2.8.3 意见与建议**

暂无

**2.9 附件**

**2.9.1 程序**

链接：https://pan.baidu.com/s/1x2gdJ3dzYI-GuMpFrc3BMw?pwd=d9tb 提取码：d9tb

**2.9.2 readme**

链接：https://pan.baidu.com/s/1x2gdJ3dzYI-GuMpFrc3BMw?pwd=d9tb 提取码：d9tb

**3 文件系统**

**3.1 实验目的**

模拟实现一个简单的类EXT2文件系统

**3.2 实验内容**

在分析 Linux 的文件系统的基础上，设计实现一个完全类似于 Ext 文件系统的虚拟多级 文件系统。

1) 可以实现下列几条命令： login ls create delete cd close read write format password

2) 列目录时要列出文件类型、文件名、创建时间、最后访问时间和修改时间。

3) 源文件可以进行读写保护。

**3.3 实验思想**

Ext2 文件系统是一个实际可用的文件系统，是庞大而复杂的。基于 Ext2 的思想和算法，设 计一个类 Ext2 的文件系统，实现 Ext2 文件系统的一个功能子集，以真正了解文件系统的运 作机制。在设计实现中，为了简化操作，可用现有操作系统的文件来代替硬盘进行硬件模拟。 1）设计文件系统应该考虑的几个层次：介质的物理结构；物理操作；设备驱动程序完成；文件系统的组织结构（逻辑组织结构）；对组织结构其上的操作；为用户使用文件系统提供的接口

对于类 Ext2 文件系统而言，由于采用硬盘文件代替模拟物理硬盘，不涉及具体的物理 设备（硬盘）的操作，也就是说不涉及介质的物理结构和设备的驱动，仅考虑三个层次，即 文件系统低层的组织结构、其上的具体操作以及为用户提供的接口命令。而类 Ext2 文件系 统的低层组织结构，实际上就是硬盘（用文件代替）的空间划分、存放信息的布局、采用的 数据结构；其上的操作就是如何管理这些空间、如何读写维护其上的信息，为了实现这些功 能，需要定义具体的操作（函数）；为了用户对文件系统使用的方便，当然要求该文件系统 应为用户提供一些接口，这些接口可以是编程接口也可以是命令接口。

2）类 ext2 文件系统的数据结构

（1）块的定义 为简单起见，逻辑块大小与物理块大小作为参考均可定义为 512 字节。由于位图只占用 一个块，因此，每个组的数据块个数以及索引结点的个数均确定为 512\*8=4096。进一步， 每组的数据容量确定为 4096\*512B=2MB。另外，类 Ext2 中，假设只有一个用户，故可以省略 去文件的所有者 ID 的域。

（2）组描述符 为简单起见，可以只定义一个组。因此，组描述符只占用一个块。同时，superblock 块 可以省略，其功能由组描述符块代替，即组描述符块中需要增加文件系统大小，索引结点的 大小，卷名等原属于 superblock 的域。由此可得组描述符的数据结构如下： struqct ext2\_group\_desc{

类型 bytes 域 释意

char[] 16 bg\_volume\_name[16]; 卷名

int 2 bg\_block\_bitmap; 保存块位图的块号

int 2 bg\_inode\_bitmap; 保存索引结点位图的块号

int 2 bg\_inode\_table; 索引结点表的起始块号

int 2 bg\_free\_blocks\_count; 本组空闲块的个数

int 2 bg\_free\_inodes\_count; 本组空闲索引结点的个数

int 2 bg\_used\_dirs\_count; 本组目录的个数

char[] 16 psw[16]; 密码

char[] 24 bg\_pad[24]; 填充(0xff)

};

合计 80 字节，只占用一个块。

（3）索引结点数据结构 由于容量已经确定，文件（即硬盘）最大即为 2MB。需要 4096 个数据块。索引结点的数据 结构中，仍然采用多级索引机制。由于文件系统总块数必然小于 4096\*2，所以只需要 13 个 二进制位即可对块进行全局计数，实际实现用 unsigned int 16 位变量，即 2 字节表示 1 个块号。

在一级子索引中，如果一个数据块都用来存放块号，则可以存放 512/2＝256 个。因此， 只使用一级子索引可以容纳最大的文件为 256\*512=128KB。需要使用二级子索引。只使用二 级子索引时，索引结点中的一个指针可以指向 256＊256 个块，即 256\*256\*512=8MB，已经 可以满足要求了。为了尽量“像”ext2，也为了简单起见，索引结点的直接索引定义 6 个， 一级子索引定义 1 个，二级子索引定义 1 个。总计 8 个指针。

索引结点的数据结构定义：

struct ext2\_inode {

类型 字节长度 域 释意

int 2 i\_mode; 文件类型及访问权限

int 2 i\_blocks; 文件的数据块个数

int 2 i\_size; 大小(字节)

time\_t 4 i\_atime; 访问时间

time\_t 4 i\_ctime; 创建时间

time\_t 4 i\_mtime; 修改时间

time\_t 4 i\_dtime; 删除时间

int 16 i\_block[8]; 指向数据块的指针

char[] 24 i\_pad[24]; 填充 1(0xff)

} 即，每个索引结点的长度为 84 字节。

其中，i\_mode 域构成一个文件访问类型及访问权限描述。即 linux 中的 drwxrwxrwx 描述。 d 为目录，r 为读控制，w 为写控制，x 为可执行标志。并且，3 个 rwx 分别是所有者(owner)， 组(group)，全局(universe)这三个对象的权限。为了简单的起见，模拟系统中，i\_mode 的 16 位如下分配。高 8 位(high\_i\_mode)，是目录项中文件类型码的一个拷贝。低 8 位

(low\_i\_mode)中的最低 3 位分别用来标识 rwx3 个属性。高 5 位不用，用 0 填充。在显示文 件访问权限时，3 个对象均使用低 3 位的标识。这样，由这 16 位，即可生成一个文件的完 整的 drwxrwxrwx 描述。

特别的，注意在 Unix 中，不带扩展名的文件定性为可执行文件。在类 Ext2 文件系统中， 凡是扩展名为.exe,.bin,.com 及不带扩展名的，都被加上 x 标识。 i\_block 域与文件大小以及数据块的关系如图 5-6 所示。 图 5-6 类 Ext2 文件系统的二级文件索引 当文件长度小于等于 512\*6=3072 字节（3KB）时，只用到直接索引。 当文件长度大于 3072 字节，并且小于等于 3KB+128KB 即 131KB 时，除使用直接索引处， 还将使用一级子索引。 当文件长度大于 131KB，并且小于 2MB（文件系统最大值）时，将开始使用二级子索引。

（4）索引结点表 由于每个索引结点大小为 64 个字节，最多有 512\*8=4096 个索引结点。故，索引结点表 的大小为 64\*4096=256KB，512 个块。为了和 Ext2 保持一致，索引结点从 1 开始计数，0 表 示 NULL。数据块则从 0 开始计数。

（5）类 Ext2 文件系统的“硬盘”数据结构 基于以上若干定义，得到类 Ext2 文件系统的“硬盘”数据结构如表 5-2 所示。 表 5-2 类 Ext2 文件系统“硬盘” 数据结构

组描述符 数据块位图 索引结点位图 索引结点表 数据块

1 block 1 block 1 block 512 block 4096 block

512 Bytes 512 Bytes 512 Bytes 256KB 2MB

整个类 Ext2 文件系统所需要的“硬盘”空间为 1+1+1+512+4096=4611 个块。共计 4611\*512bytes=2,360,832 字节＝2305.5KB＝2.25MB。

3) 目录和文件

与 Ext2 相同，目录作为特殊的文件来处理。将第 1 个索引结点指向根目录。根目录的 索引结点中直接索引域指向数据块 0。

目录体的数据结构与 Ext2 基本相同，唯一的区别在于索引节点号用 16 位来表示：

struct ext2\_dir\_entry {

Type Bytes Field 释意

int 2 inode; 索引节点

int 2 rec\_len; 目录项长度

int 2 name\_len; 文件名长度

int 2 file\_type; 文件类型(1:普通文件，2:目录…)

char[] 15 name[EXT2\_NAME\_LEN]; 文件名

char 1 dir\_pad; 填充

};

当文件系统在初始化时，根目录的数据块（即数据块 1）将被初始化。其所包含的所有 索引节点号以及目录项长度域将被置 0。当文件被删除时，其所在目录项长度不变，索引节 点号将被置 0;

当新建一个文件时，程序将从目录的数据块查找索引节点号为 0 的目录项，并检查其长 度是否够用。是，则分配给该文件，否则继续查找，直到找到长度够用，或者是长度为 0（即 未被使用过）的地址，为文件建立目录项。

当建立的是一个目录时，将其所分配到的索引结点所指向的数据块清空。并且自动写入 两个特殊的目录项。一个是当前目录“.”,其索引结点即指向本身的数据块。另一个是上一 级目录“..”，其索引结点指向上一级目录的数据块。例如，/root 目录。其索引结点号为 1。并且，第 1 个数据块存放着该目录的目录项。/root 目录在文件系统初始化时自动生成。 同时，在目录项中自动生成以下两项:

inode rec\_len name\_len file\_type name

1 8 1 2 . \0

1 9 2 2 . . \0

作为简化，文件类型使用 1 表示普通文件，2 表示目录

4）“存储空间”管理

这里涉及到类 Ext2 文件系统的 5 个底层操作：索引结点的分配与释放、数据块的分配 与释放以及数据块的寻址。

这些操作将采用 Ext2 基本相同的方法实现。区别在于：Ext2 中对 SuperBlock 的操作将 变成对组描述符的操作。此外，数据块在分配时不采取预先分配策略。查找空闲块的方法可 采用从某个起始点开始线性查找。

5）类 Ext2 文件系统的操作 作为一个类似于 Ext2 的文件系统，应该提供诸如下相关操作

操作命令 参数 功能

format 无 格式化文件系统

password 无 修改文件系统的密码

ls 无 列举当前目录的文件（目录）

create 文件类型，文件名 在当前目录创建文件(目录)

delete 文件类型，文件名 在当前目录删除文件(目录)

cd 目录名 切换目录

close 目录深度 按照目录深度关闭目录

read 文件名 从文件中读取数据

write 文件名 向文件中写入数据

6）内存中数据结构

为了实现 5）中定义的类 Ext2 文件系统的这些操作，内存中也必须有相应的数据结构。 unsigned int fopen\_table[16]; //打开文件表，暂无实现

ext2\_group\_desc group\_desc; //组描述符

ext2\_inode inode; //inode 节点

ext2\_dir\_entry dir; //目录体

unsigned int last\_allco\_inode=0; //上次分配的索引节点号

unsigned int last\_allco\_block=0; //上次分配的数据块号

7）程序结构

（1）初始化类 Ext2 文件系统 在已有的文件系统的基础上建立一个大小为 FS\_SPACE=2,360,832 字节(即 2.25MB)的文 件 vdisk，这个文件即用来模拟（代替）硬盘。以后，文件系统的所有操作，均通过读写这 个文件实现。并且，完全模拟硬盘读写方式，一次读取 1 个块，即 512 字节。即使只有 1 个字节的修改，也通过读写一个数据块来实现。，另外，常驻内存的数据结构也被初始化。

（2）文件系统级（底层）函数及其子函数 这些函数完成所有文件系统底层的操作封装，并为上层即命令层提供服务。该层实现了 所有对文件系统“硬盘”的块操作功能。例如：分配和回收索引结点与数据块、索引结点的 读取与写入、数据块的读取与写入、索引结点及数据块位图的设置、组描述符的修改以及多 级索引的实现等。

（3）命令层函函数

文件系统所支持的命令及其功能在这一层实现。一共实现 11 个命令：ls, create, delete, cd, close, read, write，format，password，login，logout。为了实现这些命 令，本层使用底层所提供的服务。

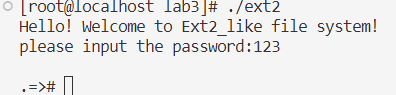
（4）用户接口层 主要功能是接收及识别用户命令，进行词法分析，提取命令及参数。组织调用命令层对 应的命令实现相应功能。本层实际上是一个基于命令层基础上的 shell。为了完善接口的功 能，shell 程序中增加退出程序的命令 exit。

**3.4 实验步骤**

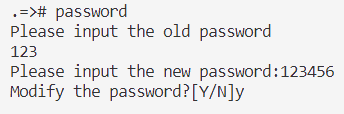
根据设计原理和相关的分析编写程序，进行调试后对结果进行分析。

**3.5 程序运行初值及结果分析**

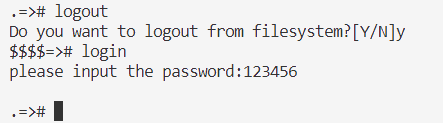
1.初始化和登入login



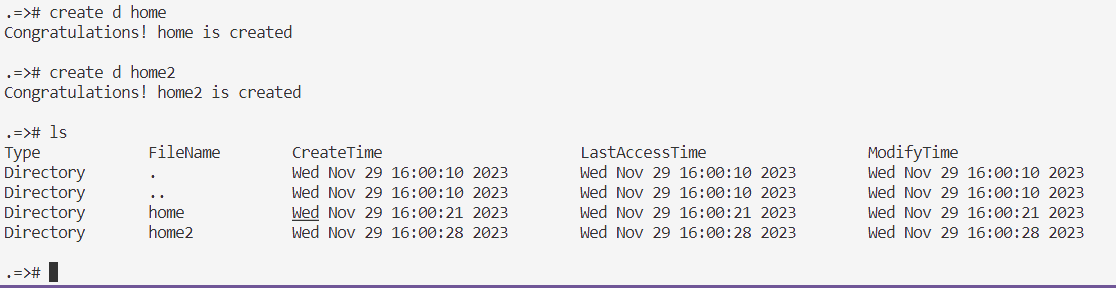
2.password修改密码

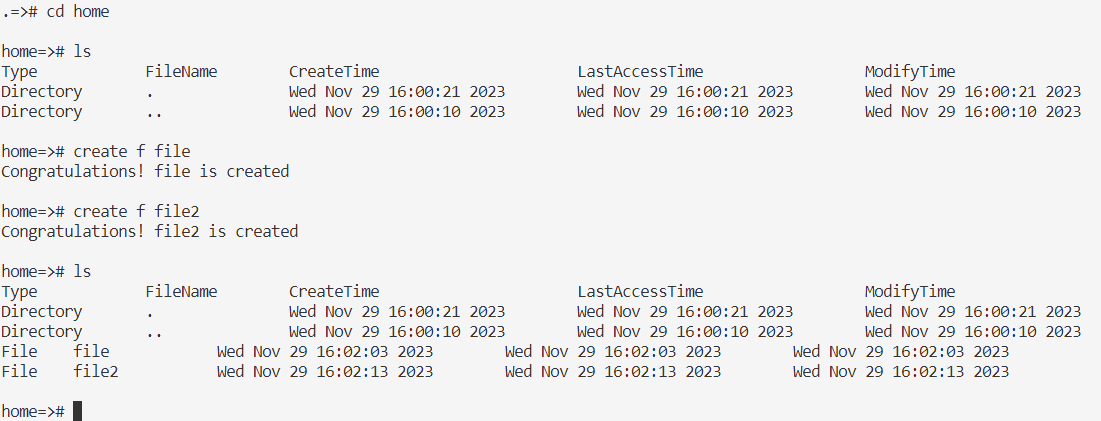


3.logout登出，并用新密码重新登入

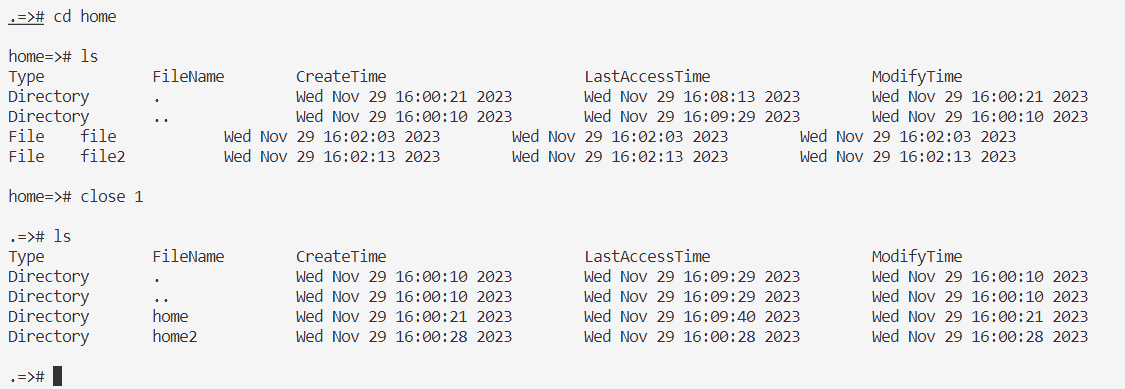


4.create创建目录/文件

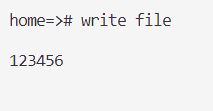




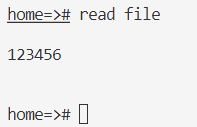
5.cd打开目录 close关闭目录



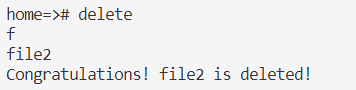
6.wirte写入文件



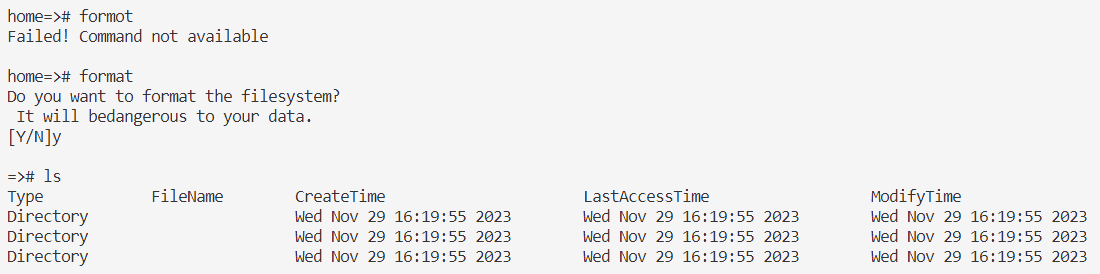
7.read读文件



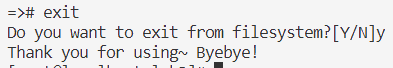
8.delete删除文件



9.format初始化



10.exit退出系统



**3.6 实验总结**

**3.6.1 遇到的问题和解决过程**

暂无

**3.6.2 实验收获**

深入理解了类EXT2文件系统的运行机制和原理，提高了编程能力

**3.6.3 意见与建议**

暂无

**3.7 附录**

**3.7.1 程序代码**

链接：https://pan.baidu.com/s/1x2gdJ3dzYI-GuMpFrc3BMw?pwd=d9tb 提取码：d9tb

**3.7.2 readme**

链接：https://pan.baidu.com/s/1x2gdJ3dzYI-GuMpFrc3BMw?pwd=d9tb 提取码：d9tb