2009 年中南大学 Linux

一、选择题(每小题 1 分，共 40 分)

1)在Linux 系统中，硬件设备大部分是安装在 \_\_\_**B**\_\_\_目录下的。

A. /mnt **B. /dev** C. /proc D. /swap

2)为了得到外壳程序中命令行参数的个数，我们可以使用变量：

**A.** **$#** B. $@ C. $0 D. $! 3)系统管理常用的二进制文件，一般放置在 \_\_\_**/sbin**\_\_\_\_\_目录下。

**A.** **/sbin** B. /root C. /usr/sbin D. /boot

4)如下所示的 /etc/fstab 文件中，表示文件系统类型的参数在第 \_\_\_\_列 /dev/had5 /mnt/dosdata **msdos** defaults, usrquota 1 2

**A.** **3** B. 4 C. 5 D. 6

在 /etc/fstab 文件中，表示文件系统类型的参数是在第 3 列。每行的格式通常如下：

<文件系统> <挂载点> <文件系统类型> <挂载选项> <dump选项> <fsck选项>

其中，第3列表示文件系统类型，例如ext4、ntfs、msdos等等。在这个例子中，文件系统类型是msdos。

5)系统当前已经加载的所有文件系统在 \_\_\_\_\_\_\_\_\_\_\_文件中得到反映。

A. /usr/sbin/cfdisk B. /sbin/fdisk **C.** **/etc/mtab** D. /etc/fstab

/usr/sbin/cfdisk：这是一个磁盘分区工具，用于对硬盘进行分区操作。它并不涉及文件系统的挂载和卸载，因此与本题无关

/sbin/fdisk：这也是一个磁盘分区工具，与cfdisk类似，用于对硬盘进行分区操作。它也不涉及文件系统的挂载和卸载，因此与本题无关

/etc/mtab：这是一个动态生成的文件，用于记录当前系统中已经加载的所有文件系统及其相关信息，例如挂载点、文件系统类型、挂载选项等。它反映的是当前系统的实际状态，因此是本题的正确答案

/etc/fstab：这是一个静态的配置文件，用于指定系统在启动时应该挂载哪些文件系统。它列出了系统中所有可能用到的文件系统，包括根文件系统、用户家目录、CD-ROM等，每个文件系统都指定了相应的挂载点、文件系统类型、挂载选项等。它是系统启动时用来自动挂载文件系统的依据，而不是反映当前系统的实际状态。

**6)通过修改文件 \_\_\_\_\_\_\_\_，可以设定开机时候自动安装的文件系统。**

A. /etc/mtab B. /etc/fastboot **C.** **/etc/fstab** D. /etc/inetd.conf 7)我们可以修改 /etc/inittab 文件中类似于 \_\_\_\_\_\_的一行，调整启动级别，使得 系统登录时采用 X Window 方式。

A. id:3:initdefault B. si::sysinit:/etc/rc.d/rc.sysinit

**C.** **x:5:respawn:/usr/bin/X11/kdm** **-nodaemon** D. ca::ctrlaltdel:/sbin/shutdown -t5 -r now

8)一个设置了 a 标志的分区，是系统中的 \_\_\_\_\_\_\_\_

A. 主分区 B. 逻辑分区 **C.** 活动分区 D. 额外分区

9)按下 \_\_\_\_\_\_\_\_ 键能终止当前运行的命令

**A.** **Ctrl-C** B. Ctrl-F C. Ctrl-B D. Ctrl-D

10) Linux 启动的第一个进程 init 启动的第一个脚本程序是 \_\_\_\_\_\_\_\_

。

A./etc/rc.d/init.d **B./etc/rc.d/rc.sysinit** C./etc/rc.d/rc5.d

D./etc/rc.d/rc3.d

11)显示用户的主目录的命令是 \_ **echo** **$HOME** \_

**A.** **echo** **$HOME** B. echo $USERDIR C. echo $ENV D. echo $ECHO

12)哪一个命令能用来删除当前目录及其子目录下名为 'core'的文件 \_\_\_**find . -name core -exec rm {} \;**\_\_

13)以下哪个运行级别是管理用的单用户模式 \_\_\_\_1\_\_\_\_

A. 0 ：停机 **B. 1：用户** C. 2：多用户 D. s

14)以下哪个命令可以**终止一个用户的所有进程**\_\_\_\_

A. skillall B. skill C. kill **D.** **killall**

15)在 Linux 下面查看磁盘使用情况用哪个命令? \_\_df\_\_

**A.** **df** B. du C. diskstat D. netstat

**16)下面关于 Shell 的说法，不正确的是 \_\_\_\_\_\_\_\_**

A.操作系统的外壳 B.用户与 Linux 内核之间的接口程序

C.一个命令语言解释器 **D.一种和 C类似的程序语言**

17 ) LILO的配置文件是 \_\_\_\_\_\_\_\_

A. /etc/conf **B.** **/etc/lilo.conf**

C. /proc/kcore D. /usr/local/

**18) 内核不包括的子系统是 \_\_\_硬件管理系统 \_\_\_\_\_**

A. 进程管理系统 B. 内存管理系统 C. I/O 管理系统 D.文件管理系统

19)关于 Linux 下的 C/C++程序的编译和运行，说法不正确的是 \_\_\_\_\_\_\_\_

A. 在编译 C++程序的时候， 用 GCC和 g++是等价的 B. g++专门用来处理 C++程序

**C.** **GCC**默认生成的可执行文件名是 **a.exe** D. C和 C++程序都可以用 GCC编译及连接

GCC 和 g++ 都可以用来编译 C++ 程序，它们的区别在于编译过程中使用的默认库不同。GCC 默认使用 C 库，而 g++ 默认使用 C++ 库。因此，使用 g++ 编译 C++ 程序会更加方便。

g++ 是 GCC 的一个前端程序，专门用于处理 C++ 程序。

GCC 可以编译和链接 C 和 C++ 程序。可以使用 gcc 命令编译 C 程序，使用 g++ 命令编译 C++ 程序。

**20) 从后台启动进程，应在命令的结尾加上符号 \_\_\_\_&\_\_\_\_**

**A.** **&** B. @ C. # D. $

二、简答题；(每小题 6 分，共 30 分)

**1)解释 i 节点在文件系统中的作用。**

在 **linux** 文件系统中，是以块为单位存储信息的，为了找到某一个文件在存储空间中存放的位置，用**i** 节点对一个文件进行索引。**I** 节点包含了描述一个文 件所必须的全部信息。所以 **i** 节点是文件系统管理的一个数据结构。

**2)简要说明符号链接与硬链接的区别。**

硬连接记录的是目标的 **inode,**符号连接记录的是目标的 **path**。符号连接就 像是快捷方式 **,**而硬连接就像是备份 **!** 符号连接可以做跨分区的 **link**；而硬连接 由于 **inode** 的缘故， 只能在本分区中做 **link.**所以 **,**符号连接的使用频率要高的多。

3)简述动态优先级多级反馈循环调度算法思想及优缺点。

4)网间进程通信主要有哪两种模式？简要说明它们的不同。

5)简要说明你学习 LINUX使用的实验环境。

**四、设计题(共 10 分)**

1、( 10 分)某系统管理员需每天做一定的重复工作，请按照下列要求，编制一 个解决方案：

( 1)读取 /xyz 目录下 x1 文件中每行第一个域 的全部数据加入到 /backup 目录下的 bak01.txt 文件内；

( 2)将/data 目录下的所有目录和文件归档并压缩为文件： backup. tar.gz；

(3)将 IDE接口的 CD-ROM卸载(假设： CD-ROM的设备名为 hdc)；

**cut -f 1 /xyz/x1 >> /backup/bak01.txt**

**tar -cv -f backup.tar.gz /data**

**umount /dev/hdc**

一 . 单项选择题 (本题 30 分， 每小题 1.5 分) 在每小题列出的四个选项中只有一 个是符合题目要求的，错选或未选均无分。

1．下面哪个 Linux 命令可以**一次显示一页内容** \_\_\_more\_\_\_

A. pause B. cat **C.** **more** D. grep

2．怎样更改一个文件的权限设置 \_\_\_chomd\_\_\_

A. attrib **B.** **chmod**C. change D. file

3．如何从当前系统中卸载一个已装载的文件系统 \_\_umount\_\_\_\_

**A.** **umount** B. dismount

C. mount –u D. 从 /etc/fstab 中删除这个文件系统项

4．在大多数 Linux 发行版本中，图形方式的运行级定义为 \_\_\_5\_\_\_

A.1 B.2 C.3 **D.5**

5．用命令 ls -al 显示出文件 ff 的描述如下所示，由此可知文件所有者所在组的 权限是 \_\_B\_\_\_\_

-rwxrw-r-- 1 root root 599 Cec 10 17:12 ff

A. rwx **B.** **rw-** C. r-- D.无法获知

**6．内核不包括的子系统是 \_\_**硬件管理系统**\_\_\_\_\_**

A. 进程管理系统 B. 内存管理系统

C. I/O 管理系统 **D.** 硬件管理系统

7．下面哪一个是 Linux 缺省状态下使用的文件系统类型: **Ext2**

**A** **.Ext2** B. NFS C. Msdos D. Vfat

8 ．Linux 文件系统的文件都按其作用分门别类地放在相关的目录中，对于外部 设备文件，一般应将其放在 \_\_\_**/dev** \_\_\_目录中

**A.** **/dev** B. /etc C. /bin D. /lib

9．关闭 linux 系统可使用命令 \_\_\_shutdown -h now\_\_\_

A.Ctrl+Alt+Del B.Ctrl+ALT **C.shutdown** **-h** **now** D. reboot

10．用命令 ls -al 显示出文件 ff 的描述如下所示， 由此可知文件 ff 的类型为 \_\_普通文件\_\_\_

-rwxr-xrwx 1 root root 599 Cec 10 17:12 ff

A. 普通文件 B. 硬链接 C. 目录 **D.** 符号链接

11．删除文件命令为 \_\_rm\_\_\_。

A .mkdir B. rmdir C. mv **D.** **rm**

12．改变文件所有者的命令为 \_\_\_chown\_\_\_。

A chmod B. touch **C.** **chown** D. cat

13．在下列命令中，不能显示文本文件内容的命令是 \_\_\_join\_\_\_。

A. more B.less C.tail **D.** **join**

A. more 和 B. less 都是用于在终端上逐页显示文本文件内容的命令。它们都可以用于查看大型文件，同时还支持搜索、定位等功能。两者的区别在于，less 比 more 更加灵活，可以向前或向后滚动，支持更多的命令和选项。因此，在大多数情况下，建议使用 less 命令。

C. tail 命令可以用于显示文件的末尾若干行，通常用于实时监视日志文件或其他正在不断写入的文件。tail 命令还支持 -f 选项，可以跟踪文件的变化并实时显示新增的内容。

D. join 命令可以用于将两个文件的行连接起来，连接的键可以是文件中某一列的值。具体来说，join 命令会将两个文件中指定列的值相同的行合并成一行。这个命令通常用于按照某些关键字合并数据文件。

14．查看系统当前进程信息的命令是 \_\_ps\_\_\_\_

**A.ps** B.df C.man D. ls

15. 进程有三种基本状态，分别是 \_\_运行态、就绪态和等待态 \_\_\_\_\_。

A. 准备态、执行态和退出态 B 精确态、模糊态和随机态

**C.** 运行态、就绪态和等待态 D 手工态、自动态和自由态

16. 从后台启动进程，应在命令的结尾加上符号 \_\_\_**&**\_\_。

**A.** **&** B. @ C. # D. $

17. 一个进程调用 fork 系统调用后，会有什么结果？C

A. 生成一个进程，从 main()处开始执行 B. 生成一个线程，从 main()开始执行

**C. 生成一个进程，从 fork 之后开始执行** D. 生成一个线程，从 fork 之后开始执行

18. 一个文件名字为 rr.Z，可以用来解压缩的命令是 \_\_ **uncompress** \_\_\_。

A. tar B. gzip C. compress **D.** **uncompress**

19. 文件权限读、写、执行的三种标志符号依次是 \_\_ **rwx** \_\_\_。

**A.** **rwx** B. xrw C. rdx D. srw

20. UNIX 操作系统属于(多用户多任务)类操作系统

A．单用户单任务 B．单用户多任务 C. 多处理机 **D.** 多用户多任务

二、填空题 (本题 15 分，每空 1 分)

21．在 Linux 系统中，以 \_\_文件\_\_\_\_方式访问设备 。

22．编写的 Shell 程序运行前必须赋予该脚本文件 \_\_执行\_\_权限。

**23．唯一标识每一个用户的是 \_\_用户名 \_\_\_\_\_和\_\_UID\_\_\_\_\_。**

24．在 Linux 系统中，用来存放系统所需要的配置文件和子目录的目录是 \_\_/etc\_\_

25．在 Linux 系统中，用来存放用户信息的文件是 \_\_/etc/passwd\_

26．结束进程的命令是 \_kill\_\_\_

27．将前一个命令的标准输出作为后一个命令的标准输入， 称之为 \_管道\_\_\_

28．管道通信包括 \_有名管道 \_和\_\_无名管道 \_\_\_\_\_两种。

**29．信号量主要是用来解决不同进程间的 \_\_同步互斥 \_\_\_\_\_问题。**

**30．进程接收到信号后可以忽略信号， 但是\_\_SIGKILL\_\_\_\_\_ 和\_\_SIGSTOP\_\_\_\_\_ 这两个信号不能被忽略。**

31 ． 安 装 Linux 系统 对硬 盘分 区时 ， 必 须有 两种 分区 类 型 ： \_\_/(root根分区)\_\_\_和 \_\_swap\_\_\_\_

三、简答题 (本题 30 分，每小题 6 分)

32．什么是符号链接，什么是硬链接？符号链接与硬链接的区别是什么？

33 ．Linux 的进程间通信机制有哪些？请分别简要说明。

**#** 无名管道 **(** **pipe** **)**：管道是一种半双工的通信方式，数据只能单向流动，而且 只能在具有亲缘关系的进程间使用。进程的亲缘关系通常是指父子进程关系。 **#** 有名管道 **(named** **pipe)** ： 有名管道也是半双工的通信方式，但是它允许无 亲缘关系进程间的通信。

**#** 信号量 **(** **semophore** **)** ： 信号量是一个计数器， 可以用来控制多个进程对共享 资源的访问。它常作为一种锁机制，防止某进程正在访问共享资源时，其他进

程也访问该资源。因此，主要作为进程间以及同一进程内不同线程之间的同步

手段。

**#** 消息队列 **(** **message** **queue** **)**： 消息队列是由消息的链表，存放在内核中并由 消息队列标识符标识。消息队列克服了信号传递信息少、管道只能承载无格式 字节流以及缓冲区大小受限等缺点。

**#** 信号 **(** **signal** **)** ： 信号是一种比较复杂的通信方式，用于通知接收进程某个 事件已经发生。

**#**共享存储器 **(** **shared** **memory)**：共享内存就是映射一段能被其他进程所访问的 内存，这段共享内存由一个进程创建，但多个进程都可以访问。共享内存是最 快的 **IPC** 方式，它是针对其他进程间通信方式运行效率低而专门设计的。它往 往与其他通信机制，如信号量，配合使用，来实现进程间的同步和通信。

**#** 套接字 **(** **socket** **)** ： 套解口也是一种进程间通信机制， 与其他通信机制不同的

是，它可用于不同及其间的进程通信。

**35．简要说明里 linux 环境下编译、运行 C 程序的基本工具和过程。 工具： gcc、 make；gdb**

如果程序很小(源文件很少)，可以直接用 gcc编译

如果程序较大 (源文件比较多) ，则要先编写程序的 Makefile 文件， 然后用 make

编译

如果要debug：使用gdb；

四、编程与应用题 (本题 25 分)

37．阅读下面 shell 程序，请写出执行结果。( 6 分)

echo -e "Please enter the score:"

while read SCORE

do

case $SCORE in

?|[1-5]? ) echo "Failed!"

echo "Please enter the next score:";;

6?) echo "Passed!"

echo "Please enter the next score:";;

7?) echo "Medium!"

echo "Please enter the next score:";;

8?) echo "Good!"

echo "Please enter the next score:";;

9?|100) echo "Great!"

echo "Please enter the next score:";;

\*) exit;;

esac

done

其中输入的数据为

62 Passed!

Please enter the next score

80 Good!

5 Failed!

1000

38．用 Shell 编程，判断一个文件是不是字符设备文件，如果是将其拷贝到 /dev

目录下，否则提示 “This file is not exit的信息。” ( 6 分)

**#!/bin/bash**

**filename="test"**

**if** **[** **-c** **"$filename"** **]**

**then**

**cp** **"$filename"** **/dev**

**else**

**echo** **"This** **file** **is** **not** **exit"**

**fi**

39．编写一段 C 语言程序使其完成：父进程创建两个子进程，父子进程都在屏

幕上显示自己的进程 ID 号。要求先显示子进程的 ID 号， 后显示父进程的 ID 号。

(6 分)

**#include** **<sys/type.h>**

**#include** **<sys/wait.h>**

**#include** **<unistd.h>**

**#include** **<stdio.h>**

**#include** **<stdlib.h>**

**int** **main()** **{**

**pid\_t** **pid1,** **pid2;**

**pid1** **=** **fork();**

**if(pid1** **==** **0)** **{**

**printf("** 子进程 **1** ：**%d\n",** **getpid());**

**exit(0);**

**}**

**else** **{**

**wait(NULL);**

**pid2** **=** **fork();**

**if(pid2** **==** **0)** **{**

**printf("** 子进程 **2**：**%d\n",** **getpid());**

**exit(0);**

**}**

**else** **{**

**wait(NULL);**

**printf("** 父进程： **%d\n",** **getpid());**

**}**

**}**

**return** **0;**

**}**

40．编写利用 IPC 实现进程通信的 C 程序。该程序主要模拟根据帐号查询余额 的过程。包括三方面：

1)请求进程从标准输入读入帐号，并将该帐号通过消息队列发送给服务进程；

2)服务进程接收该帐号后，按照请求的先后顺序在标准输入上输入该帐户的姓 名和余额，并将结果返回给请求进程；

3)请求进程接收返回的信息，并将结果输出在标准输出上。服务进程先于请求

进程启动， 请求进程启动时要携带请求编号，可同时启动多个请求进程。 ( 7 分)

/\* 请求方程序 msgreq.c\*/

**#include<sys/types.h>**

**#include<sys/ipc.h>**

**#include<sys/msg.h>**

**#include<stdio.h>**

**static** **struct** **msgbuf** **{**

**long** **mtype;**

**char** **mtext[256];**

**}** **sndbuf,** **rcvbuf,** **\*msgp**

**extern** **int** **errno;**

**int** **main(int** **argc,** **char** **\*\*argv)** **{**

**int** **rtrn,** **msqid**

**char** **name[10];**

**double** **balance;**

**if** **(argc** **!=** **2)**

**{**

**fprintf(stderr,** **"msgreq** **[01-99]\n");**

**exit(-1);**

**}**

**if** **(** **(msqid** **=** **msgget(0x888,** **IPC\_CREAT** **|** **0660))** **==** **-1** **)**

**{**

**fprintf(stderr,** **"msgget** **888** **failed** **!\n");**

**exit(-1);**

**}**

**msgp** **=** **&sndbuf;**

**sprintf(sndbuf.mtext,** **"%2.2s",** **argv[1]);**

**printf("** 输入 **4** 位帐号 **:");**

**scanf("%s",** **&sndbuf.mtext[2]);**

**sndbuf.mtext[6]** **=** **0;**

**msgp->mtype** **=** **666;**

**rtrn** **=** **msgsnd(msqid,** **msgp,** **strlen(sndbuf.mtext),** **0);**

**if** **(rtrn** **==** **-1)**

**{**

**perror("msgsnd");**

**exit(-1);**

**}**

**msgp** **=** **&rcvbuf;**

**fprintf(stderr,** **"**等待后台数据处理进程的回答 **....");**

**rtrn** **=** **msgrcv(msqid,** **msgp,** **100,** **atoi(argv[1]),** **0);**

**if(rtrn** **==** **-1)**

**{**

**perror("msgrcv");**

**exit(-1);**

**}**

**sscanf(rcvbuf.mtext,** **"%[^|]|%lf",** **name,** **&balance);**

**printf("\n** 姓名 **=%s\n",** **name);**

**printf("** 余额**=%lf\n",** **balance);**

**}**

/\* 服务方程序 msgcenter.c\*/

**static** **struct** **msgbuf1** **{**

**long** **mtype;**

**char** **mtext[100];**

**}** **sndbuf,** **rcvbuf** **,** **\*msgp;**

**extern** **int** **errno;**

**int** **main()** **{**

**int** **rtrn,** **msgqid**

**char** **strbuf[100];**

**if** **(** **(msqid** **=** **msgget(0x888,** **IPC\_CREAT** **|** **0600))** **==** **-1** **)** **{**

**fprintf(stderr,** **"msgget** **888** **failed** **!\n");**

**exit(-1);**

**}**

**while(1){**

**msgp** **=** **&rcvbuf;**

**fprintf(stderr,** **"** 等待前台进程的请求 **....");**

**rtrn** **=** **msgrcv(msqid,** **msgp,** **100,** **666** **,** **MSG\_NOERROR);**

**if(rtrn** **==** **-1){**

**perror("msgrcv");**

**exit(-1);**

**}**

**msgp** **=** **&sndbuf;**

**sprintf(strbuf,** **"%2.2s\0",** **rcvbuf.mtext);**

**msgp->mtype** **=** **atoi(strbuf);**

**printf("\n** 输 入 帐 号 **=%4.4s** 的 帐 户 姓 名 **:",** **&rcvbuf.mtext[2]);** **scanf("%s",** **sndbuf.mtext);**

**strcat(sndbuf.mtext,** **"|");**

**printf("** 输入该帐户余额 **:");**

**scanf("%s",** **strbuf);**

**strcat(sndbuf.mtext,** **strbuf);**

**rtrn** **=** **msgsnd(msqid,** **msgp,** **strlen(sndbuf.mtext),** **0);** **if** **(rtrn** **==** **-1)** **{**

**perror("msgsnd");**

**exit(-1);**

**}**

**}**

**}**