## Linux课堂作业

**姓名：胡海龙 学号：201510414304 班级：软工2015级3班**

**一、使用命名管道实现文本传输**

代码如下：

#include <stdio.h>

#include <fcntl.h>

#include <string.h>

#include <stdlib.h>

#include <sys/select.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <errno.h>

int main()

{

int i, rfd,wfd,len=0,fd\_in;

char str[32];

int flag,stdinflag;

fd\_set write\_fd,read\_fd;

struct timeval net\_timer;

mkfifo("fifo1",S\_IWUSR|S\_IRUSR|S\_IRGRP|S\_IROTH); /\*mkfifo 函数创建命名管道\*/

mkfifo("fifo2",S\_IWUSR|S\_IRUSR|S\_IRGRP|S\_IROTH); /\*mkfifo 函数创建命名管道\*/

wfd =open("fifo1",O\_WRONLY); /\*以写方式打开管道文件\*/

rfd =open("fifo2",O\_RDONLY); /\*以只读方式打开管道文件\*/

if(rfd<=0||wfd<=0)return 0;

printf("这是发送端!\n");

while(1){

FD\_ZERO(&read\_fd); /\*清除一个文件描述符集\*/

FD\_SET(rfd,&read\_fd); /\*将文件描述符rfd 加入文件描述符集read\_fd \*/

FD\_SET(fileno(stdin),&read\_fd);

net\_timer.tv\_sec=5;

net\_timer.tv\_usec=0;

memset(str,0,sizeof(str)); /\*memset 函数初始化清空\*/

if(i=select(rfd+1, &read\_fd,NULL, NULL, &net\_timer) <= 0)

continue;

if(FD\_ISSET(rfd,&read\_fd)){

read(rfd,str,sizeof(str)); /\*读取管道，将管道内容存入str 变量\*/

printf("----------------------------\n");

printf("one:%s\n",str); /\*打印输出str 变量内容\*/

}

if(FD\_ISSET(fileno(stdin),&read\_fd)){

printf("----------------------------\n");

fgets(str,sizeof(str),stdin);

len=write(wfd,str,strlen(str)); /\*写入管道\*/

}

}

close(rfd);

close(wfd);

}

#include <stdio.h>

#include <fcntl.h>

#include <string.h>

#include <stdlib.h>

#include <sys/select.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <errno.h>

int main()

{

int i, rfd,wfd,len=0,fd\_in;

char str[32];

int flag,stdinflag;

fd\_set write\_fd,read\_fd;

struct timeval net\_timer;

mkfifo("fifo1",S\_IWUSR|S\_IRUSR|S\_IRGRP|S\_IROTH); /\*mkfifo 函数创建命名管道\*/

mkfifo("fifo2",S\_IWUSR|S\_IRUSR|S\_IRGRP|S\_IROTH); /\*mkfifo 函数创建命名管道\*/

rfd =open("fifo1",O\_RDONLY); /\*以只读方式打开管道文件\*/

wfd =open("fifo2",O\_WRONLY); /\*以写方式打开管道文件\*/

if(rfd<=0||wfd<=0)return 0;

printf("这是接收端!\n");

while(1){

FD\_ZERO(&read\_fd); /\*清除一个文件描述符集\*/

FD\_SET(rfd,&read\_fd); /\*将文件描述符rfd 加入文件描述符集read\_fd \*/

FD\_SET(fileno(stdin),&read\_fd);

net\_timer.tv\_sec=5;

net\_timer.tv\_usec=0;

memset(str,0,sizeof(str)); /\*memset 函数初始化清空\*/

if(i=select(rfd+1,&read\_fd,NULL, NULL, &net\_timer) <= 0)

continue;

if(FD\_ISSET(rfd,&read\_fd)){

read(rfd,str,sizeof(str)); /\*读取管道，将管道内容存入str 变量\*/

printf("----------------------------\n");

printf("one:%s\n",str); /\*打印输出str 变量内容\*/

}

if(FD\_ISSET(fileno(stdin),&read\_fd))

{

printf("----------------------------\n");

fgets(str,sizeof(str),stdin);

len=write(wfd,str,strlen(str)); /\*写入管道\*/

}

}

close(rfd);

close(wfd);

}

**二、消息队列实现聊天程序**

代码如下：

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<unistd.h>

#include<sys/ipc.h> //文件预处理，包含ftok,msgget,msgsnd等函数

#include<sys/msg.h> //文件预处理，包含msgget,msgsnd等函数

#include<signal.h> //文件预处理，包含进程通信函数库

#define KEYPATH "."

#define PROJECT\_ID 'a' //宏定义

struct msg{ //结构体，定义消息结构

long mtype; //此程序中为group代码，相同mtype者可以进行聊天

char name[10]; //聊天者名字

char mtext[100]; //聊天内容

};

void fun\_ctrl\_c(); //自定义信号处理函数声明

int main(void)

{

int msgpid,len;

key\_t key;

pid\_t pid1;

char buf[100],sender[20];

struct msg msgs,msgr;

(void)signal(SIGINT,fun\_ctrl\_c); //如果按了ctrl+c键，调用fun\_ctrl\_c函数

if((key=ftok(KEYPATH,PROJECT\_ID))==-1){ //调用ftok函数产生标准的key

perror("Error in creating the key:");

exit(1);

}

if((msgpid=msgget(key,IPC\_CREAT|0666))==-1){ //调用msgget函数，创建打开消息队列

perror("Error in creating the message queue:");

exit(1);

}

printf("Please enter your Group Number:"); //输入组号码

scanf("%ld",&msgs.mtype);

getchar();

printf("Please enter your name:"); //输入姓名

scanf("%s",msgs.name);

getchar();

printf("\nYou entered HYL chat room!\n");

sleep(1);

system("clear");

printf("You can enter what you want to say~\n");

pid1=fork(); //调用fork函数产生子进程

if(pid1<0){ //子进程产生失败，退出

perror("Error:");

exit(1);

}

else if(pid1==0){ //子进程

while(1){ //无限循环，接收队列信息

len=msgrcv(msgpid,&msgr,sizeof(msgr),msgs.mtype,0); //接受相同mtype的队列消息

if(len<0){

perror("Read msg failed:");

exit(1);

}

if(len==0) continue;

if((strcmp(msgr.name,msgs.name))==0){ //如果接收的为本人发送的信息，重新发送

msgsnd(msgpid,&msgr,100,0);

continue;

}

printf("From %s: %s",msgr.name,msgr.mtext); //输出队列消息

}

}

else{ //父进程

while(1){

if((fgets(msgs.mtext,100,stdin))==NULL){

printf("You cannot sent the empty msg!\n");

continue;

}

if((msgsnd(msgpid,&msgs,100,0))<0){ //输入信息，发送

perror("msg send failed:");

}

else{

if(strcmp(msgs.mtext,"886...\n")==0) fun\_ctrl\_c(); //如果发送”886…”，进入退出函数

printf("%s: %s",msgs.name,msgs.mtext);

}

}

}

}

void fun\_ctrl\_c() //退出函数{

char a;

printf("Are you sure to quit?(y?)\n"); //询问是否退出

scanf("%c",&a);

getchar();

if(a=='y'){

printf("Thank you for using HYL chat room!\n");

sleep(1);

system("clear");

exit(0); //是，则退出}

else{

printf("Back to HYL chat room.\n"); //否，则返回

return;

}

}

**三、mmap实现简单聊天程序**

代码如下：

/\*7-11.c 程序：匿名内存映射\*/

#include<sys/types.h> /\*文件预处理，包含waitpid、kill、raise 等函数库\*/

#include<unistd.h> /\*文件预处理，包含进程控制函数库\*/

#include <sys/mman.h>

#include <fcntl.h>

typedef struct /\*结构体，定义一个people 数据结构\*/

{

char name[4];

int age;

}people;

main(int argc, char\*\* argv) /\*C 程序的主函数，开始入口\*/

{

pid\_t result;

int i;

people \*p\_map;

char temp;

p\_map=(people\*)mmap(NULL,sizeof(people)\*10,PROT\_READ|PROT\_WRITE,

MAP\_SHARED|MAP\_ANONYMOUS,-1,0); /\*调用mmap 函数，匿名内存映射\*/

result=fork(); /\*调用fork 函数，复制进程,返回值存在变量result 中\*/

if(result<0) /\*通过result 的值来判断fork 函数的返回情况，这儿进行出错处理\*/

{

perror("创建子进程失败");

exit(0);

}

else if (result==0) /\*返回值为0 代表子进程\*/

{

sleep(2);

for(i = 0;i<5;i++)

printf("子进程读取: 第 %d 个人的年龄是： %d\n",i+1,(\*(p\_map+i)).age);

(\*p\_map).age = 110;

munmap(p\_map,sizeof(people)\*10); /\*解除内存映射关系\*/

exit(0);

}

else /\*返回值大于0 代表父进程\*/

{

temp = 'a';

for(i = 0;i<5;i++)

{

temp += 1;

memcpy((\*(p\_map+i)).name, &temp,2);

(\*(p\_map+i)).age=20+i;

}

sleep(5);

printf( "one:%d\n",(\*p\_map).age );

printf("解除内存映射……\n");

munmap(p\_map,sizeof(people)\*10);

printf("解除内存映射成功！\n");

}

}