**实训日志**

|  |  |  |
| --- | --- | --- |
| 学校：宝鸡文理学院 | 专业：电子科学与技术 | 学号：201691074036 |
| 日期：2018/9/5 | 星期：星期三 | 姓名：刘超飞 |
| 内容：（课程主要内容，代码/作业，收获，感悟，建议等）   1. 课程主要内容   文件操作  1.文件操作是什么  如何对磁盘中的文件进行操作  文件操作是操作系统提供给开发者操作磁盘文件提供的一系列接口。  接口：API(应用程序开发接口)  API说白了是一系列的函数  2.文件操作的分类  标准I/O：标准库提供的对文件操作的接口(跨平台，速度慢)  文件I/O：类Unix操作系统提供的操作文件的接口(只适用于类Unix操作系统)  类Unix操作系统：  Unix操作系统：  Linux操作系统：Ubuntu、CenterOS、Redhet  3.文件I/O  文件IO如何去操作文件：  1.文件IO是系统调用(是操作系统提供的操作硬件资源的接口)  系统调用的作用：由操作系统统一调度稀有资源为用户提供服务。  从磁盘中读取文件内容：  标准I/O：fread  文件I/O：read    4.文件I/O常用的函数  如何新建一个文件，往文件中写入内容：    文件描述符：文件描述符是一个大于等于0的一个整数  我们可以通过这个整数去操作文件  几个特殊的文件描述符：  0：标准输入流  1：标准输出流  2：标准错误流  使用open函数获取文件描述符：  open函数：  /\*需要添加的头文件\*/  #include <sys/types.h>  #include <sys/stat.h>  #include <fcntl.h>  /\*  \*函数名：opne  \*函数功能：打开文件  \*函数参数：  \* const char \*pathname：带路径的文件名  \* int flags：打开的方式：O\_RDONLY(只读), O\_WRONLY(只写), orO\_RDWR(可读可写).  \*函数返回值：int：成功返回文件描述符，失败返回-1  \*/  int open(const char \*pathname, int flags);  /\*  \*函数名：opne  \*函数功能：打开文件，如果文件不存在，则创建文件  \*函数参数：  \* const char \*pathname：带路径的文件名  \* int flags：打开的方式：O\_RDONLY(只读), O\_WRONLY(只写), orO\_RDWR(可读可写) O\_CREAT(文件不存在时创建)  mode\_t mode:mode是一个八进制的数，代表想要创建的文件权限  \*函数返回值：int：成功返回文件描述符，失败返回-1  \*/  int open(const char \*pathname, int flags, mode\_t mode);  如何查看文件的权限：ls  ls：查看当前路径下文件  ls -a:查看当前路径下所有文件（包括隐藏文件）  ls -l:详细列举文件的信息  文件类型和权限  文件类型 文件权限  - rw-rw-r--  文件类型：-(普通文件)  d(目录)  c(字符设备)  b(块设备)  p(管道文件)  s(socket文件)  l(连接文件)  文件权限：  r：可读  w：可写  x：可执行  -：没有该权限    当前用户的权限 当前用户组的权限 其他用户的权限  rw- rw- r--  注意：在编程中，我们使用3位八进制数来表示文件权限：  0664 rw-rw-r--  rw- rw- r--  110 110 100  打开文件：  创建文件：  open("test", O\_WRONLY | O\_CREAT, 0664);  close函数：  关闭文件描述符：  /\*需要添加的头文件\*/  #include <unistd.h>  /\*  \*函数名：close  \*函数功能：关闭文件描述符  \*函数参数：int fd：想要关闭的文件描述符  \*函数返回值：int：成功返回0，失败返回-1；  \*/  int close(int fd);  实现touch命令：  ./a.out filename  创建文件  main函数传参：  /\*  \*int agrc：参数个数包含命令本身  \*char \*argv[]：argv是一个数组，数组的每一个元素是一个char \*，每一个元素对应main函数参数的首地址  \*/  int main(int argc, char \*argv[])  {  return 0;  }  字符串是特殊字符串数组(以0为结尾)  write函数：  /\*头文件\*/  #include <unistd.h>  /\*  \*函数名：write  \*函数功能：往文件描述符中写内容  \*函数参数：  \* int fd：想要被写入的文件描述符  \* const void \*buf:想要被写的内容的内存首地址  \* size\_t count:想要写入的字节数  \*函数返回值：ssize\_t:成功返回写入的字节数，失败返回-1  \*/  ssize\_t write(int fd, const void \*buf, size\_t count);  read：从文件中读取内容  /\*头文件\*/  #include <unistd.h>  /\*  \*函数名：read  \*函数功能：是从文件描述符中读取内容  \*函数参数：  \* int fd：被读取的文件描述符  \* void \*buf:存储被读出来的内容的内存空间的首地址  \* size\_t count:想要读出的字节数  \*函数返回值：ssize\_t:成功返回读到的字节数，失败返回-1，当read的返回值为0时，就读到了文件末尾。  \*/  ssize\_t read(int fd, void \*buf, size\_t count);  cat file  man 2 open  man read  man 2 write  man close  memset：  /\*头文件\*/  #include <string.h>  /\*  \*函数名：memset  \*函数功能：以字节为单位填充内存，按照固定字符填充  \*函数参数：  \* void \*s：被填充的内存的首地址  \* int c：想要填充的字符  \* size\_t n：想要填充的字节数  \*函数返回值：void \*：返回被填充的内存区域的首地址，即s  \*/  void \*memset(void \*s, int c, size\_t n);  lseek:文件读写指针重定位  /\*头文件\*/  #include <sys/types.h>  #include <unistd.h>  /\*  \*函数名：lseek  \*函数功能：文件读写指针重定位  \*函数参数：  \* int fd：想重定位读写指针的文件的文件描述符  \* off\_t offset：偏移量(基于基准的)  \* int whence：基准  \*函数返回值：off\_t：返回偏移量(基于文件开头)  \*/  off\_t lseek(int fd, off\_t offset, int whence);  基准: SEEK\_SET:文件头  SEEK\_CUR:当前位置  SEEK\_END:文件末尾  如何求文件长度：  int len = lseek(fd, 0, SEEK\_END);   1. 作业   #include <stdio.h>  #include <unistd.h>  #include <sys/types.h>  #include <sys/stat.h>  #include <fcntl.h>  int main()  {  int fd=-1;  fd=open("1.jpg",O\_RDWR);  if(fd<0)  {  puts("open error");  return -1;  }  int ret=-1;  char buf[10]={0};  ret=read(fd,buf,10);  if(ret ==10)  {  int i=0,temp;  for(i=0;i<5;i++)  {  temp=buf[i];  buf[i]=buf[9-1];  buf[9-i]=temp;  }  lseek(fd,0,SEEK\_SET);  write(fd,buf,10);    }  close(fd);  return 0;  }  #include <unistd.h>  #include <sys/types.h>  #include <sys/stat.h>  #include <fcntl.h>  #include <string.h>  int main(int argc,char \*argv[])  {  if(argc!=3)  {  puts("usage:./a.out filename");  return -1;  }  int fd=-1;  int cd=-1;  fd=open(argv[1],O\_RDONLY);  cd=open(argv[2],O\_WRONLY | O\_CREAT,0664);  if(fd<0)  {  puts("creat file error");  return -1;  }  char buf[1001];  memset(buf,0,1001);  int ret=0;  while(1)  {  memset(buf,0,1001);  ret=read(fd,buf,1000);  if(ret>0)  {  write(cd,buf,strlen(buf));  }  else if(ret==0)  {  break;  }  }  close(fd);  return 0;  }   1. 收获和感受   今天学习了用c程序实现文件的一系列操作，今天程序的结构挺简单的，理解起来相对容易。作业上面则有点难度，作业是在今天内容上的一个提高，做起来挺累了，早点洗洗睡了........ | | |