

**课 程 实 验 报 告**

**课程名称： 操 作 系 统 原 理**

**专业班级： 计算机科学与技术1609**

**学 号： U201614758**

**姓 名： 梅朝瑞**

**指导教师： 谢 夏**

**报告日期： 2019/1/1**

**计算机科学与技术学院**

# 实验四：Linux文件目录

## 实验目的

1. 了解Linux文件系统与目录操作；
2. 了解Linux文件系统目录结构；
3. 掌握文件和目录的程序设计方法。

## 实验内容

1. **程序要求**

编程实现目录查询功能：

• 功能类似ls -lR；

• 查询指定目录下的文件及子目录信息；

显示文件的类型、大小、时间等信息；

• 递归显示子目录中的所有文件信息。

1. **运行环境**

**软件配置：**

主机：Windows 10 专业版 Build 17763.195

虚拟机：VMware Workstation 14 Pro，Deepin 15.8 Linux/GNU

开发环境：Eclipse IDE for C/C++ Developers Oxygen Release (4.7.0)

**硬件：**

AMD Ryzen 5 1600 Six-Core Processor 3.20 GHz

Kinston HyperX Fury DDR4 2400 8G

NVIDIA GeForce GTX 1060 6GB

1. **源程序**

#include <unistd.h>

#include <unistd.h>

#include <sys/stat.h>

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <dirent.h>

#include <string.h>

#include <time.h>

#include <pwd.h>

void getRWX(const unsigned short m);

void printdir(char \*dir, int depth){

DIR \*dp;

struct dirent \*entry;

struct stat statbuf;

if ((dp = opendir(dir)) ==NULL){

printf("fail to open the dir");

return;

}

chdir(dir);

while((entry = readdir(dp))!=NULL){

//以该目录项的名字为参数,调用lstat得到该目录项的相关信息;

lstat(entry->d\_name,&statbuf);

if(S\_ISDIR(statbuf.st\_mode)){

if(strcmp(entry->d\_name,"..")==0||strcmp(entry->d\_name,".")==0)

continue;

// 打印目录项的深度、目录名等信息

// 递归调用printdir,打印子目录的信息,其中的depth+4;

getRWX(statbuf.st\_mode);

printf("%d ",depth); //深度

struct passwd \*pwd;

pwd = getpwuid(statbuf.st\_uid);

printf("%s ",pwd->pw\_name); //文件所有者ID

struct passwd \*gpwd;

if((gpwd = getpwuid(statbuf.st\_gid))!=NULL)

{

printf("%s\t",gpwd->pw\_name); //文件所有者组ID

}

printf("%ld\t",statbuf.st\_size); //文件大小

printf("%s\t",strtok(ctime(&statbuf.st\_mtime)+4,"\n")); //最后修改内容时间

printf("%s\n",entry->d\_name);

printdir(entry->d\_name,depth+4);

}else{

getRWX(statbuf.st\_mode);

printf("%d ",depth); //深度

struct passwd \*pwd;

pwd = getpwuid(statbuf.st\_uid);

printf("%s ",pwd->pw\_name); //文件所有者ID

struct passwd \*gpwd;

if((gpwd = getpwuid(statbuf.st\_gid))!=NULL)

{

printf("%s\t",gpwd->pw\_name); //文件所有者组ID

}

printf("%ld\t",statbuf.st\_size); //文件大小

printf("%s\t",strtok(ctime(&statbuf.st\_ctime)+4, "\n")); //最后修改内容时间

printf("%s\n",entry->d\_name);

}

}

chdir("..");

closedir(dp);

}

void getRWX(const unsigned short m)

{

if (m& S\_IRUSR)

printf("r");

else

printf("-");

if (m & S\_IWUSR)

printf("w");

else

printf("-");

if (m & S\_IXUSR)

printf("x");

else

printf("-");

if (m & S\_IRGRP)

printf("r");

else

printf("-");

if (m & S\_IWGRP)

printf("w");

else

printf("-");

if (m & S\_IXGRP)

printf("x");

else

printf("-");

if (m & S\_IROTH)

printf("r");

else

printf("-");

if (m & S\_IWOTH)

printf("w");

else

printf("-");

if (m & S\_IXOTH)

printf("x ");

else

printf("- ");

return;

}

int main(){

char dirr[50];

printf("please input the dir name\n");

scanf("%s",dirr);

printdir(dirr,1);

return 0;

}

1. **实验结果**

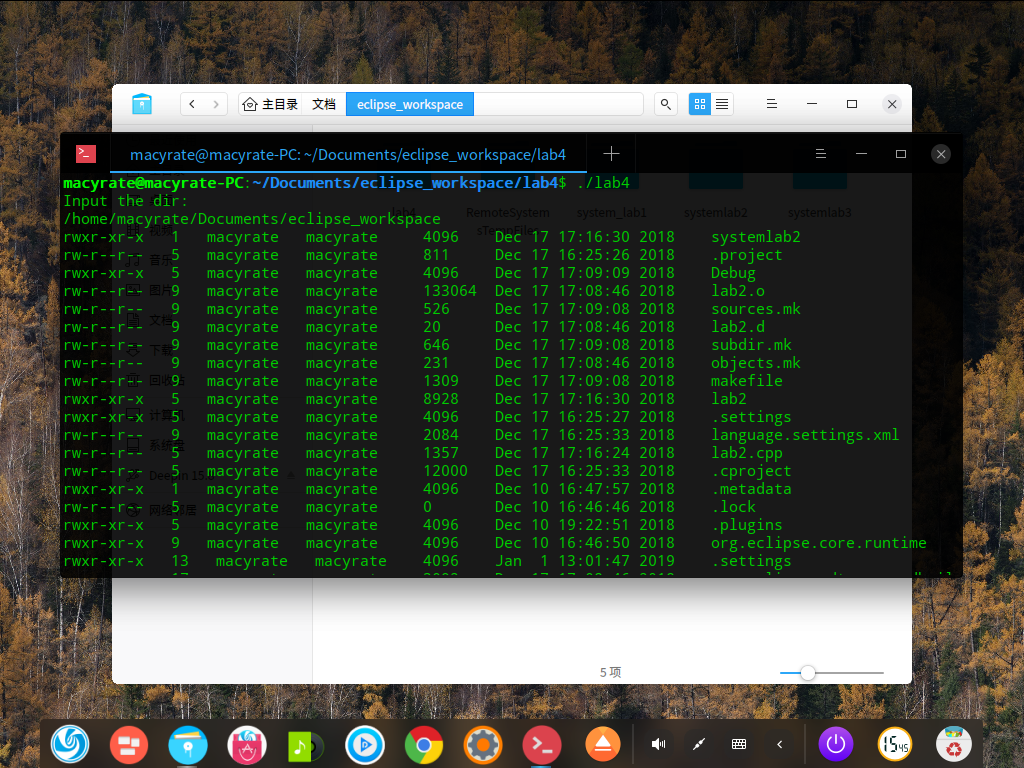


图4-1 程序开始执行

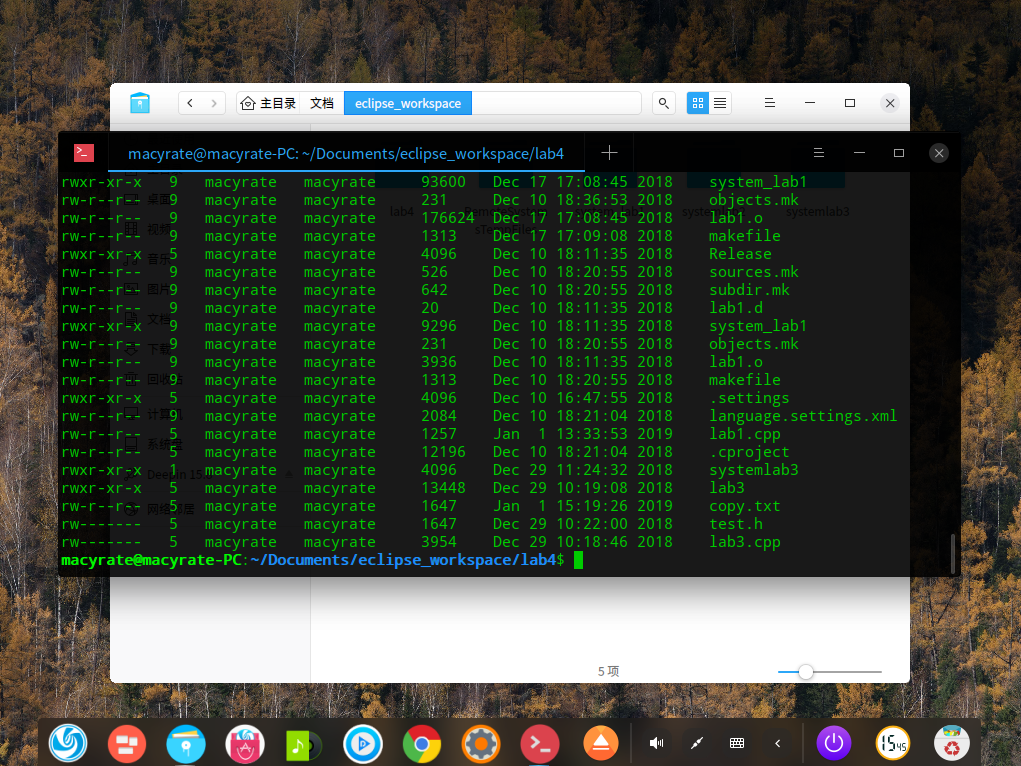


图4-2 程序结束执行

## 实验心得

这是操作系统的第四次实验。

反而，是四次实验中最为简单的一个，因为文件属性和目录结构等功能Linux都给出了接口，文档中甚至还给出了参考程序结构，可以说基本是娱乐向实验。不过，在纵观实验文档之后，Linux系统关于文件管理的功能实在令人赞叹，简洁凝练而又十分实用，充满功能性，命令行环境下虽没有Windows那样的“直观”，但ls指令的数种附带参数的形式有简有繁，比起图形界面更有几分计算机的“理性感”。

所以，唯一麻烦的只是如何组织这些接口所给出的信息，好好规划一番，使得它们能规整地在终端上显示出来。这方面说到底其实也并不复杂，毕竟理想的结果样例已经给出，只要“依葫芦画瓢”即可。

通过这次实验，我了解了Linux文件系统与目录操作、文件系统的目录结构，掌握了文件和目录的程序设计方法，收获很大。