**电 子 科 技 大 学**

作 业 报 告

# 学生姓名：王冉恒 学 号：202422900232 指导教师：李林学生E-mail：[1976569841@qq.com](mailto:djhuang_1@qq.com)

**一、作业名称**

自学stat等函数，获取文件元数据信息。

# 二、作业要求

实现“ls -l”的基本功能

至少能输出：

文件类型

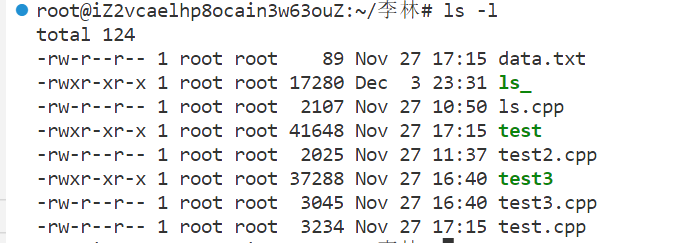
9个权限位信息

文件大小

文件名称

# 三、设计与实现

ls -l 是一个用于列出目录内容的 Linux 命令，其功能是以长格式（long format）显示目录中的文件和子目录信息。它会显示文件或目录的详细信息，包括权限、所有者、所属组、大小、创建日期和时间等。这些信息以列的形式呈现，每一行代表一个文件或目录。



显示内容的意义如下：

第一列显示文件的权限和文件类型。 第二列显示与该文件相关的硬链接数。

第三列和第四列分别显示文件的所有者和所属组。

第五列显示文件的大小（以字节、千字节或以其他合适的单位显示）。第六列、第七列和第八列显示文件的创建日期和时间。

最后一列显示文件或目录的名称。

此作业实现了输出文件类型、权限位信息、文件大小、文件名称等功能。具体实现如下：

1. 文件大小

输入当前文件路径后，函数会递归的给出当前路径下所拥有的所有除隐藏文件外的文件的路径信息，对于每个文件使用stat函数打开，将其文件信息存储在一个struct stat结构体中，通过输出这个结构体的st\_size成员，即可得到这个文件的大小。

1. 文件名称

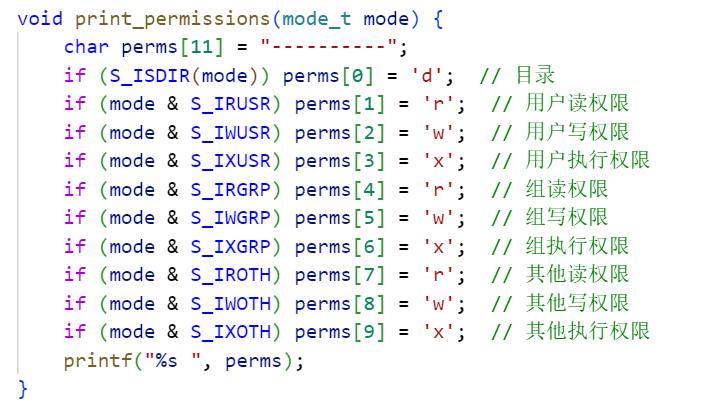
输入当前文件路径后，调用readdir函数，即可得到执行当前文件夹下第一个文件的指针，这个指针是一个结构体指针，其成员d\_name即是这个文件的名字，使用while循环，不停的调用readdir函数，移动指针，即可得到所有文件的名称。当指针指向的文件是一个目录文件时，就会递归的调用这个获取文件名的函数，以此来得到当前文件子文件夹下的文件。

1. 文件类型与权限位

对于每个文件使用stat函数打开，将其文件信息存储在一个struct stat结构体中，这个结构体含有一个变量st\_mode,st\_mode 主要包含了 3 部分信息：  
a. 15bit ~ 12bit 保存文件类型  
b. 11bit ~ 9bit 保存执行文件时设置的信息  
c. 8bit ~ 0bit 保存文件访问权限

使用st\_mode & 掩码就可以得到文件的类型及其权限。

具体操作：



# 四、测试

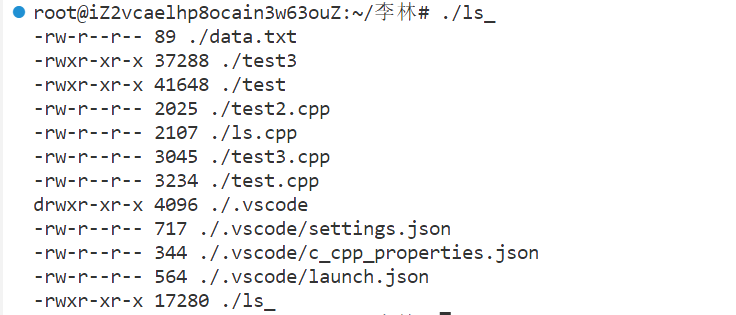
编译：

g++ -o ls\_ ls.cpp

执行：

./ls

输出：



本处的实现是自动输出当前文件下所有文件（除隐藏文件外）的信息，所以无需输入参数，执行脚本，自动就会输出当前文件夹下所有文件的文件大小、文件名称、文件权限等信息。

# 五、附录

#include <stdio.h>

#include <stdlib.h>

#include <sys/stat.h>

#include <unistd.h>

#include <string.h>

#include <dirent.h>

void print\_permissions(mode\_t mode) {

char perms[11] = "----------";

if (S\_ISDIR(mode)) perms[0] = 'd'; *// 目录*

if (mode & S\_IRUSR) perms[1] = 'r'; *// 用户读权限*

if (mode & S\_IWUSR) perms[2] = 'w'; *// 用户写权限*

if (mode & S\_IXUSR) perms[3] = 'x'; *// 用户执行权限*

if (mode & S\_IRGRP) perms[4] = 'r'; *// 组读权限*

if (mode & S\_IWGRP) perms[5] = 'w'; *// 组写权限*

if (mode & S\_IXGRP) perms[6] = 'x'; *// 组执行权限*

if (mode & S\_IROTH) perms[7] = 'r'; *// 其他读权限*

if (mode & S\_IWOTH) perms[8] = 'w'; *// 其他写权限*

if (mode & S\_IXOTH) perms[9] = 'x'; *// 其他执行权限*

printf("%s ", perms);

}

void print\_file\_info(const char \*path) {

struct stat file\_stat;

if (stat(path, &file\_stat) == -1) {

perror("stat");

return;

}

*// 输出文件类型和权限*

print\_permissions(file\_stat.st\_mode);

*// 输出文件大小*

printf("%ld ", file\_stat.st\_size);

*// 输出文件名称*

printf("%s\n", path);

}

void list\_files(const char \*path) {

DIR \*dir;

struct dirent \*entry;

if (!(dir = opendir(path))) {

perror("opendir");

return;

}

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

char full\_path[1024];

if (strcmp(entry->d\_name, ".") == 0 || strcmp(entry->d\_name, "..") == 0) *//不输出隐藏文件*

continue; *//跳过接下来的步骤，进入下一次循环*

snprintf(full\_path, sizeof(full\_path), "%s/%s", path, entry->d\_name);*//不会打印输出，将后面字符串的值存到full\_path中*

print\_file\_info(full\_path);

struct stat statbuf;

if (stat(full\_path, &statbuf) == -1) {

perror("stat");

continue;

}

if (S\_ISDIR(statbuf.st\_mode)) { *//如果是目录，则进入递归循环*

list\_files(full\_path);

}

}

closedir(dir);

}

int main() {

list\_files(".");

return 0;

}