Nikeem Dunkelly-Allen, ECE322, Assignment 4

myls A picture containing text

Description automatically generated

myls -l option

Text

Description automatically generated

Myls -o option

Text

Description automatically generated

Myls –p option

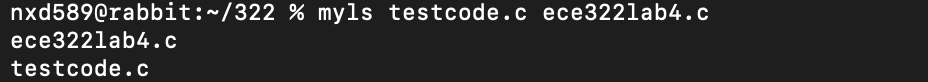


myls -sop option

A picture containing text

Description automatically generated

myls with some files



myls with a file that doesn’t exist

Graphical user interface, text

Description automatically generated

Myls more options

Text

Description automatically generated

Myls more options again -im

Text

Description automatically generated

Myls -iom

A picture containing calendar

Description automatically generated

Text

Description automatically generated

#include <stdio.h>

#include <dirent.h>

#include <errno.h>

#include <stdlib.h>

#include <string.h>

#include <time.h>

#include <sys/stat.h>

#include <pwd.h>

#include <sys/types.h>

#include <unistd.h>

void mylsnormal(const char \*dir)

{

struct dirent \*\*d;

int n;

int i = 0;

n = scandir(dir, &d,NULL,alphasort);

if (n < 0){

perror ("scandir");

}

else{

while (i<n)

{

//not listing hidden files

if (d[i]->d\_name[0] == '.'){

i++;

continue;

}

if (d[i]->d\_type == 4){ directories

printf("%s(D)\n", d[i]->d\_name);

i++;

continue;

}

printf("%s\n", d[i]->d\_name);

i++;

} }

}

void mylswopt(const char \*dir, const char \*options[])

{

struct dirent \*\*d;

int n;

int i = 0;

char buf[10];

char \*temp;

if (strcmp(dir, ".") != 0){

strcpy(buf, "./");

strcat(buf, dir);

int l = chdir(buf);

if (l < 0){

perror("chdir");}}

n = scandir(".", &d,NULL,alphasort);

if (n < 0){

perror ("scandir");

}

else{

while (i<n)

{

if (d[i]->d\_name[0] == '.'){

i++;

continue;

}

struct stat s;

int r = stat(d[i]->d\_name, &s);

if (r < 0){

perror("Stat");

fprintf(stderr, "Can't stat %s\n", d[i]->d\_name);

exit(EXIT\_FAILURE);

}

if (d[i]->d\_type == 4){

printf("%s(D), ", d[i]->d\_name);}

else{

printf("%-15s ", d[i]->d\_name);}

int size = s.st\_size;

int uid = s.st\_uid;

struct passwd \* x = getpwuid(uid);

int inode\_num = s.st\_ino;

int prot = s.st\_mode;

time\_t lastmod = s.st\_mtime;

time\_t created = s.st\_ctime;

char \*p = (char\*)(options[1] + 1);

while(\*p){

if (\*p == 's'){

printf("Size:%-10d", size);

}

if (\*p == 'o'){

printf("Owner:%-10s ", x->pw\_name);

}

if (\*p == 'i'){

printf("Inode:%d ", inode\_num);

}

if (\*p == 'm'){

printf("Last Modified:%.24s ", ctime(&lastmod));

}

if (\*p == 'c'){

printf("Created:%-10s ", ctime(&created));

}

if (\*p == 'r'){

if(d[i]->d\_type == 4){

printf("\n");

mylswopt(d[i]->d\_name, options);

chdir("..");

}

}

if (\*p == 'p'){

printf("Protections:");

printf( (S\_ISDIR(s.st\_mode)) ? "d" : "-");

//user

printf( (s.st\_mode & S\_IRUSR) ? "r" : "-");

printf( (s.st\_mode & S\_IWUSR) ? "w" : "-");

printf( (s.st\_mode & S\_IXUSR) ? "x" : "-");

//group

printf( (s.st\_mode & S\_IRGRP) ? "r" : "-");

printf( (s.st\_mode & S\_IWGRP) ? "w" : "-");

printf( (s.st\_mode & S\_IXGRP) ? "x" : "-");

//others

printf( (s.st\_mode & S\_IROTH) ? "r" : "-");

printf( (s.st\_mode & S\_IWOTH) ? "w" : "-");

printf( (s.st\_mode & S\_IXOTH) ? "x" : "-");

}

p++;

}

printf("\n");

i++;

}}

}

void mylsfilenormal(const char \*dir, const char \*nof[], int filenum)

{

struct dirent \*\*d;

int n;

int i = 0;

int prints =1;

n = scandir(dir, &d,NULL,alphasort);

if (n < 0){

perror ("scandir");

}

else{

while (i<n)

{

int count = 1;

if (d[i]->d\_name[0] == '.'){

i++;

continue;

}

while(count <= filenum){

if (strcmp(d[i]->d\_name, nof[count]) ==0){

if (d[i]->d\_type == 4){

printf("%s(D)\n", d[i]->d\_name);

}

else{

printf("%s\n", d[i]->d\_name);}

count++;

prints++;

continue;}

count++;

}

i++;

if (i == n && prints < count){

printf("Files not displayed, do not exist\n");

exit(EXIT\_FAILURE);

}}

}

}

void mylsfilewop(const char \*dir, const char \*list[], int filenum)

{

struct dirent \*\*d;

int n;

int i = 0;

char buf[10];

char \*temp;

int checks = 0;

int count = 2;

if (strcmp(dir, ".") != 0){

strcpy(buf, "./");

strcat(buf, dir);

int l = chdir(buf);

if (l < 0){

perror("chdir");}}

n = scandir(".", &d,NULL,alphasort);

if (n < 0){

perror ("scandir");

}

else{

while(count-1 <= filenum){

i =0;

checks++;

while (i<n)

{

if (d[i]->d\_name[0] == '.'){

i++;

continue;

}

if (strcmp(d[i]->d\_name, list[count]) !=0){

i++;

continue;}

else{

struct stat s;

int r = stat(d[i]->d\_name, &s);

if (r < 0){

perror("Stat");

fprintf(stderr, "Can't stat %s\n", d[i]->d\_name);

exit(EXIT\_FAILURE);

}

if (d[i]->d\_type == 4){

printf("%s(D), ", d[i]->d\_name);}

else{

printf("%s, ", d[i]->d\_name);}

int size = s.st\_size;

int uid = s.st\_uid;

struct passwd \* x = getpwuid(uid);

int inode\_num = s.st\_ino;

int prot = s.st\_mode;

time\_t lastmod = s.st\_mtime;

time\_t created = s.st\_ctime;

char \*p = (char\*)(list[1] + 1);

while(\*p){

if (\*p == 's'){

printf("Size:%-10d ", size);

}

if (\*p == 'o'){

printf("Owner:%-10s ", x->pw\_name);

}

if (\*p == 'i'){

printf("Inode:%d ", inode\_num);

}

if (\*p == 'm'){

printf("Last Modified:%.24s ", ctime(&lastmod));

}

if (\*p == 'c'){

printf("Created:%.24s ", ctime(&created));

}

if (\*p == 'r'){

if(d[i]->d\_type == 4){

printf("\n");

mylswopt(d[i]->d\_name, list);

chdir("..");

}

}

if (\*p == 'p'){

printf("Protections:");

printf( (S\_ISDIR(s.st\_mode)) ? "d" : "-");

printf( (s.st\_mode & S\_IRUSR) ? "r" : "-");

printf( (s.st\_mode & S\_IWUSR) ? "w" : "-");

printf( (s.st\_mode & S\_IXUSR) ? "x" : "-");

printf( (s.st\_mode & S\_IRGRP) ? "r" : "-");

printf( (s.st\_mode & S\_IWGRP) ? "w" : "-");

printf( (s.st\_mode & S\_IXGRP) ? "x" : "-");

printf( (s.st\_mode & S\_IROTH) ? "r" : "-");

printf( (s.st\_mode & S\_IWOTH) ? "w" : "-");

printf( (s.st\_mode & S\_IXOTH) ? "x" : "-");

}

p++;

}

count++;

if (count-1 > filenum){

printf("\n");

return;

}

}

printf("\n");

i++;

}

if (checks > count){

printf("Files not displayed, do not exist\n");

exit(EXIT\_FAILURE); }

}

}

}

int main(int argc, const char \*argv[])

{

if (argc == 1)

{

mylsnormal(".");

}

else if (argc > 1)

{

if (argv[1][0] == '-' && argc ==2)

{

mylswopt(".",argv);

}

else if(argv[1][0] == '-' && argc >2){

mylsfilewop(".", argv, argc-2);

}

else {

mylsfilenormal(".", argv, argc-1);

}

}

return 0;

}