**Files and Directories**

**Lab no# 08**

****

**Fall 2021**

**CSE-302 System Programming Lab**

Submitted by: **Ashfaq Ahmad**

Registration No: **19PWCSE1795**

Class Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Engr: Ma’am Madeha sheer**

**February** 17, 2021

**Department of Computer Systems Engineering**

**University of Engineering and Technology, Peshawar**

**Objective(s):**

* To know how to interact with files and directories.
* To know how to access different information about file or directory.

**Task # 01: Implement ls command.**

**Code:**

#include <stdio.h>

#include <unistd.h>

#include <dirent.h>

#include <stdlib.h>

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

{

DIR \*dirp; //return of opendir in success case is a pointer of DIR data type pointing first entry of directory.

//dirp=directory pointer

struct dirent \*direntp; //read function return pointer which point struct dirent.

//direntp=direcotry entries pointer.

//dirent has two members d\_name and d\_ino.

if(argc==1) //no command line argument. 1=this file itself

{

dirp=opendir("."); //in argc=1 case we open current working directory in which this file itself is present.

}

else if(argc==2) //one command line argument + this file itself =2.

{

dirp=opendir(argv[1]);

}

else //more than two arguments it's give error.

{

printf("Sorry! invalid No of arguments\n");

exit(0); //or return -1;

}

if(dirp==NULL) //if opendir failed it return NULL

{

printf("Sorry! directory couldn't opened Successfully\n");

perror("Reason");

return -1;

}

while((direntp=readdir(dirp))!=NULL)

//after calling readdir the name and inode no of file pointed by dirp are stored in dirent structre.

{

//direntp=readdir(dirp); we cannot use it here b/c while loop will not run..

//when we declare direntp pointer it point a garbage value.

if(direntp->d\_name[0]=='.') //d\_name is a array of character data type.

//if first character of name is (.) it will not display it.

//this if statement is actually removing the two links (.) and(..).

{

continue;

}

printf("%s ",direntp->d\_name); //print names of all directory contents.

}

printf("\n");

int x=closedir(dirp); //close directory.

if(x==-1)

{

printf("Sorry! Directory can't closed Successfully\n");

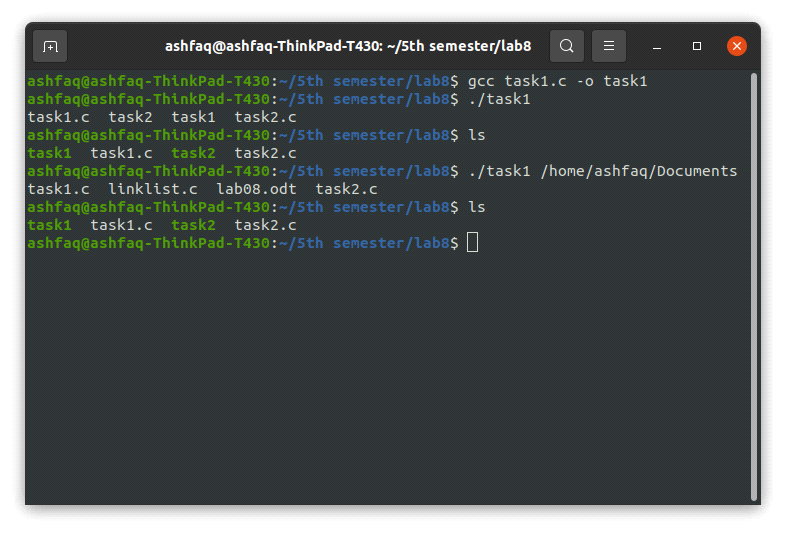
perror("Reason");

return -1;

}

}

**Output:**



**Task # 02: Implement ls -l command.**

**Code:**

#include <stdio.h>

#include <unistd.h>

#include <dirent.h>

#include <stdlib.h>

#include <sys/stat.h>

#include <pwd.h>

#include <time.h>

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

{

DIR \*dirp;

struct dirent \*direntp; //return pointer pointing first entry of directory

if(argc==1)

{

dirp=opendir(".");

}

else if(argc==2)

{

dirp=opendir(argv[1]);

int cd=chdir(argv[1]);

//stat function work only current woring directory so if we give a path of any other

//we will also need to be change the pwd according to the path.

if(cd==-1)

{

printf("Sorry! directory can't changed successfully\n");

perror("Reason");

return -1;

}

}

else //if we input more than two argument it don't give error

{

printf("Sorry! invalid No of arguments\n");

return -1;

}

if(dirp==NULL)

{

printf("Sorry! directory can't opened successfully\n");

perror("Reason");

return -1;

}

struct stat buffer;

while((direntp=readdir(dirp))!=NULL) //read untill null entry of directory reach.

{

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

{

continue;

}

int ret=stat(direntp->d\_name,&buffer);

if(ret==-1)

{

printf("Sorry! there is problem with stat function\n");

perror("Reason");

return -1;

}

if(S\_ISDIR(buffer.st\_mode))

//from st\_mode we can check the file either directory or file using micro S\_ISDIR.

{

printf("d");

}

else

printf("-");

if(S\_IRUSR&buffer.st\_mode)

{

printf("r");

}

else

printf("-");

if(S\_IWUSR&buffer.st\_mode)

{

printf("w");

}

else

printf("-");

if(S\_IXUSR&buffer.st\_mode)

{

printf("x");

}

else

printf("-");

if(S\_IRGRP&buffer.st\_mode)

{

printf("r");

}

else

printf("-");

if(S\_IWGRP&buffer.st\_mode)

{

printf("w");

}

else

printf("-");

if(S\_IXGRP&buffer.st\_mode)

{

printf("x");

}

else

printf("-");

if(S\_IROTH&buffer.st\_mode)

{

printf("r");

}

else

printf("-");

if(S\_IWOTH&buffer.st\_mode)

{

printf("w");

}

else

printf("-");

if(S\_IXOTH&buffer.st\_mode)

{

printf("x");

}

else

printf("-");

printf(" %ld",buffer.st\_nlink); //print no of links

struct passwd \*USER=getpwuid(buffer.st\_uid);

//we can get user name from passwd structure using getpwuid function having argument st\_uid.

//getpwuid return pointer to an object of struct passwd.

//struct passwd contain all necessory information regarding user login including user name.

if(USER==NULL)

{

printf("Sorry! An error occured with getpwuid\n");

perror("Reason");

return -1;

}

printf(" %s",USER->pw\_name); //print pw\_name member of struct passwd

struct passwd \*GROUP=getpwuid(buffer.st\_gid);

//also we can get group name from passwd structure using getpwuid function having argument st\_gid.

if(GROUP==NULL)

{

printf("Sorry! An error occured with getpwuid\n");

perror("Reason");

return -1;

}

printf(" %s",GROUP->pw\_name); //print pw\_name member of struct passwd

printf(" %ld ",buffer.st\_size); //print sizee of file.

//if we print "buffer.st\_mtime" directly it print time in integer form. which

//is not human understandable.

//so we convert it to string form using ctime function which is human understandale.

//The ctime() function converts the time value pointed to by time to local time in

//the form of a character string.

//ctime function take argument time of time\_t data type from struct stat and

//return a pointer to a string representing the localtime based on the argument timer.

//return string format is Www Mmm dd hh:mm:ss yyyy

char \*time=ctime(&buffer.st\_mtime);

//by refrence pass mean changes by ctime() will occure in time.

int i;

for(i=0; time[i]!='\n';i++)

;

//time array consist of 26 characters. last one is "\n" to avoid that one, we will

//run above for loop having no body upto \n when it reached we will replace it by \0.

time[i]='\0';

//instead of running above loop we can also directrly write time[24]='\0';

printf(" %s",time);

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

}

return 0;

}

**Output:**

