Operating System Code Assignment Deadline 7th November 11:59PM

# Question 1 [5 marks]

Write a program to illustrate “ls” command using system calls

1. Start.
2. Open directory Using opendir( ) system call.
3. Read the directory Using readdir( ) systemcall.
4. Print dp.name and dp.inode .
5. Repeat above step Until end of directory.
6. End

Read manual pages of the system calls used in the above pseudocode to understand what it does and what arguments does it use for what purpose.

Ans:

Code:

#include<stdio.h> #include<dirent.h> main()

{

char drnam[20];

DIR\*ptr;

struct dirent \*dtr;

printf("Enter the name of the directory \n"); scanf("%s",drnam);

ptr=opendir(drnam); if(ptr==NULL)

{

perror("directory not found"); exit(-1);

}

while(dtr=readdir(ptr)) printf("%s\n",dtr->d\_name);

printf("%s\n",dtr->d\_inode);

}

# Question 2 [10 marks]

What are links in file system scenario. Search and explore it. What commands are used to make hard and soft links. What is the difference between both the links? You have to submit the commands and their output via screenshots.

Ans:

Link is a utility program in UNIX which builds up a hard link starting with one directory then onto the next directory.

Hard Link: The utilization of a hard link permits numerous filenames to be related with a similar record since a hard link focuses to the inode of a given document, the information of which is put away on plate.

we can think a hard link as an extra name for a current document. Hard links are partner at least two record names with the equivalent inode . You can make at least one hard links for a solitary record. Hard links can't be made for catalogs and documents on an alternate filesystem or segment.

Soft links:A soft link is something like an alternate route in Windows. It is a roundabout pointer to a record or directory. In contrast to a hard link, a representative link can point to a record or a directory on an alternate filesystem or parcel.

Example:

Soft Link:

$ echo "welcome to a program in unix" > info.file

now view the data written isdide info.file.

$ cat info.file

welcome to a program in unix

as you can see that, the info.file has been created.

Now, create the a symbolic or soft link to the info.file.

To do so, run the following commands:

$ ln -s info.file softlink.file

now, compare the data of both info.file and softlink.file.

$ cat info.file

welcome to a program in unix

$ cat softlink.file

welcome to a program in unix

Hard Link:

$ echo "welcome to a program in unix" > info.file

now, verify the contents of the file.

$ cat info.file

welcome to a program in unix

The info.file has been created now.

Now, let us create the hard link to the info.file as shown below.

$ ln info.file hardlink.file

# Question 3 [10 marks]

Write a program that will perform the following command

tail -5 alpha.txt | grep ee | sort

Ans: Code:

#include <stdio.h> #include <string.h> #include <unistd.h> #include <sys/types.h>

int main(void){

pid\_t childpid; int fd1[2];

int fd2[2];

char readbuff[100]; int readb;

pipe(fd1); pipe(fd2);

printf("Before fork"); childpid = fork(); printf("After fork");

if(childpid!=0)//Parent

{

printf("1");

dup2(fd1[1], STDOUT\_FILENO);

close(fd1[0]);

execlp("tail","tail","-5", "alpha.txt", NULL);

}

else//child

{ printf("2"); childpid=fork(); if(!childpid)

{

printf("3"); readb=read(fd1[0],readbuff,sizeof(readbuff)); write(fd2[1],readbuff,sizeof(readb)); close(fd1[0]); dup2(fd2[0],STDIN\_FILENO);

execlp("sort","sort","-r", NULL);

}

else

{

printf("4"); dup2(fd2[0],STDIN\_FILENO);

}

return 0;

}

execlp("grep","grep","ee",NULL);

}

The end.