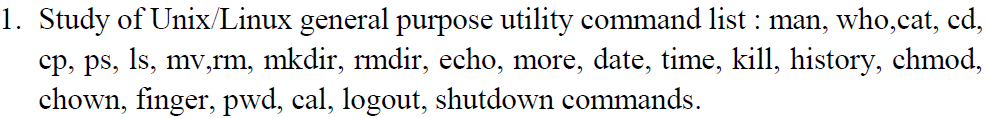
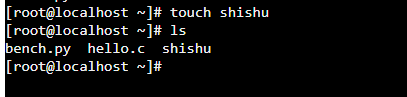
**MNNIT ALLAHABAD**

**OPERATING SYSTEM**

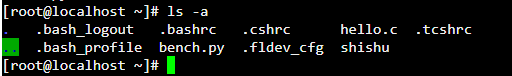
|  |  |
| --- | --- |
| **SUBMITTED BY :**  **NAME : SHISHU**  **ROLL/REG ID : 2020CA089**  **SUBMIT DATE : 17/11/2021** | **SUBMITTED TO :**  **TEACHER \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **DEPPT : COMPUTER SCIENCE**  **DEADLINE: 17/11/2021** |

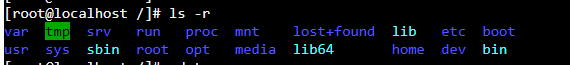
**Assignment 1**.

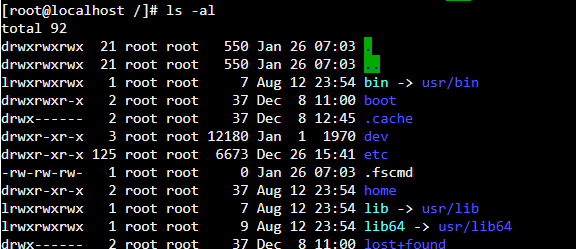


touch :- The touch is a command-line utility that is basically used to create a new empty files and update the timestamps of existing files and directories. 

ls :- ls command is use for list all file

ls –a :- In linux hidden files start with .(dot) symbol and they are not visible in regular directory. The (ls –a) command will enlist the whole list of the current directory including the hidden files. 

ls –R :- Display directory Recurcive. 

ls –al :- ls –al show in list format with details. 

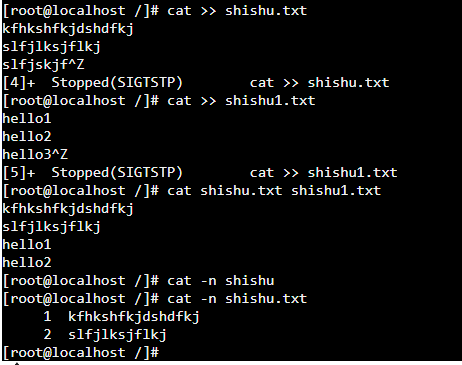
cat :- cat (concatenate) command is very frequently used in linux. It reads data from the file and gives their content as output. It helps us to create, view, concatenate file. So let us see some frequent used cat command.

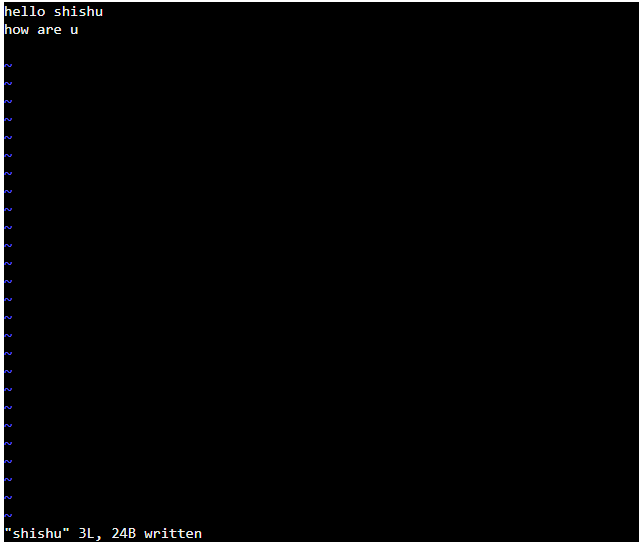
1.cat [optin] [file]…

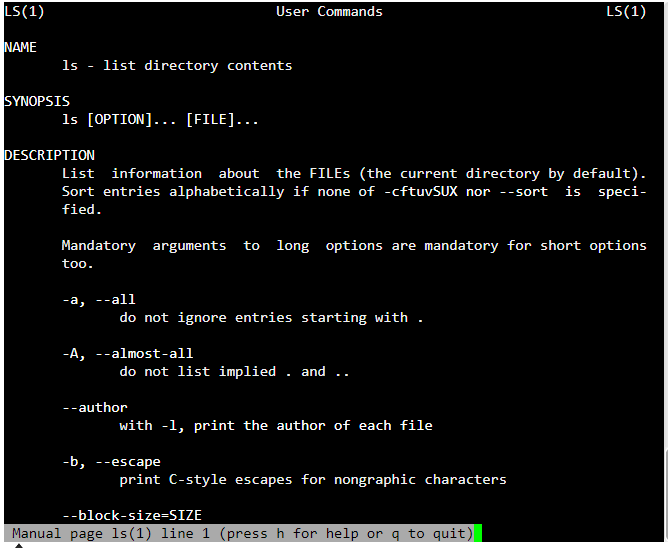
2. cat [file] | more

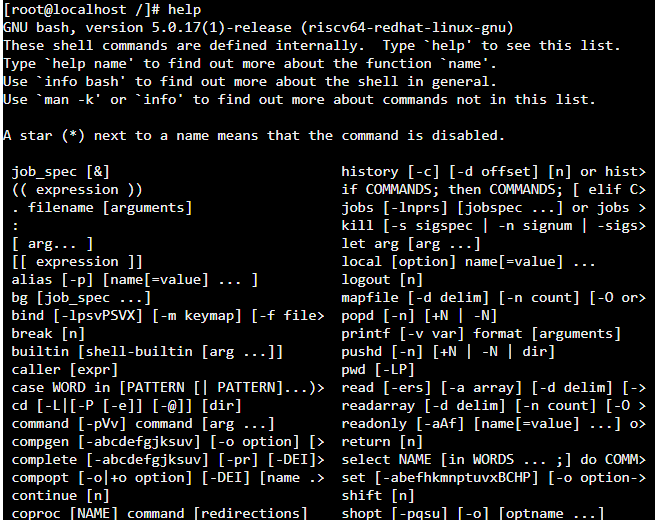
3. cat [file] | less

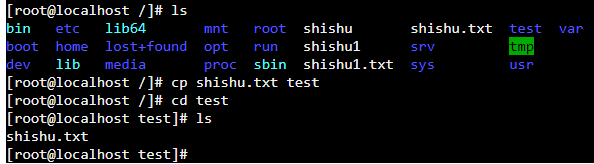
4. cat –n



vi :- There are many ways to edit files in Linux Editing files using the screen-oriented txt editor vi one of the best ways. This editor enables you to edit lines in context with other lines in the file. 

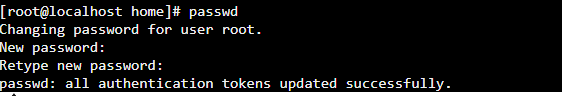
man :- It is used to display the user manual of any command that we can run on the terminal. It provides a detailed view of the command. 

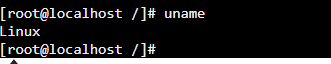
help :- help command as told before just displays information about shell built-in cammands 

cp :- It is used to copy file one destination to another destination. 

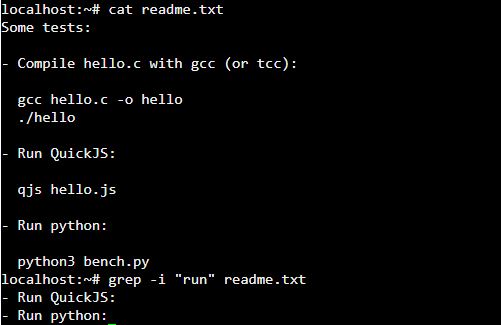
echo hello world : This command display a line of test/string on standard output or a file.



passwd :- This command is used to changing current password for user root.

uname :- uname is a command line utility that prints basic information about the operating system name and system hardare 

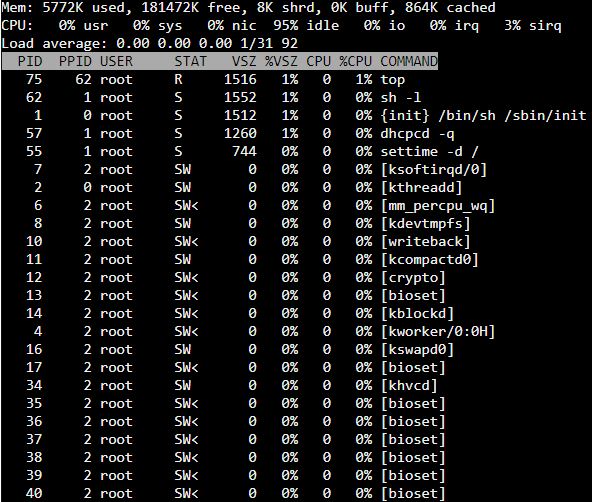
grep :- It stands for global regular expression print. The grep filter searches a file for a particular pattern of characters, and displays all lines that contain that pattern. The pattern that is searched in the file is referred to as the regular expression.



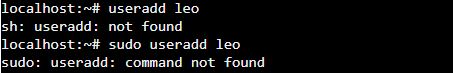
find :-The find command in UNIX is a command line utility for walking a file hierarchy. It can be used to find files and directories and perform subsequent operations on them. It supports searching by file, folder, name, creation date, modification date, owner and permissions. By using the ‘-exec’ other UNIX commands can be executed on files or folders found.

last :-This command is used to display the list of all the users logged in and out since the file **/var/log/wtmp**was created. One or more usernames can be given as an argument to display their login in (and out) time and their host-name.

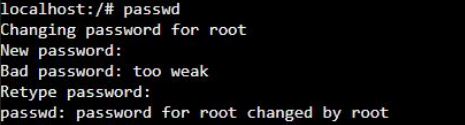


top :-This command is used to show the Linux processes. It provides a dynamic real-time view of the running system. Usually, this command shows the summary information of the system and the list of processes or threads which are currently managed by the Linux Kernel.

useradd :-This command in Linux that is used to add user accounts to your system. It is just a symbolic link to adduser command in Linux and the difference between both of them is that useradd is a native binary compiled with system whereas adduser is a Perl script which uses useradd binary in the background.

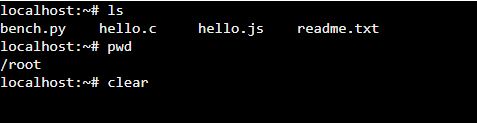


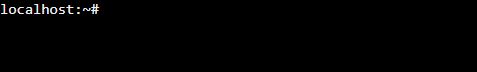
passwd :-passwd command in Linux is used to change the user account passwords. The root user reserves the privilege to change the password for any user on the system, while a normal user can only change the account password for his or her own account.



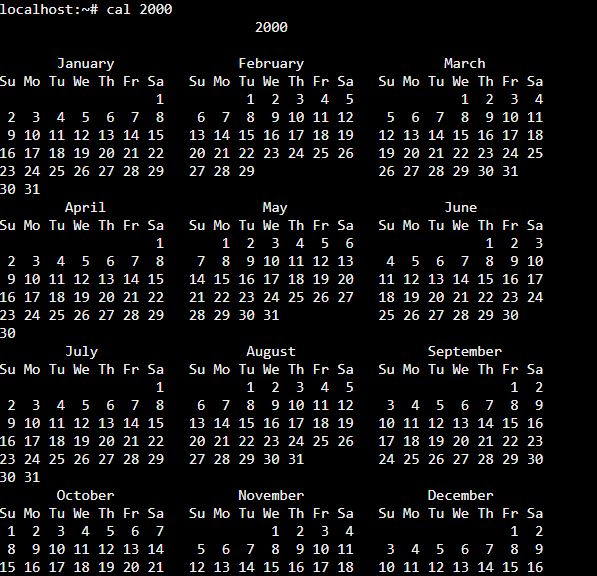
userdel :-This command in Linux system is used to delete a user account and related files. This command basically modifies the system account files, deleting all the entries which refer to the username LOGIN. It is a low-level utility for removing the users.

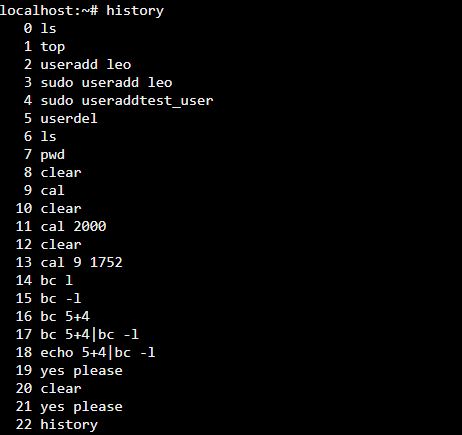


clear :- **clear**is a standard Unix computer operating system command that is used to clear the terminal screen.



cal 2000 :- By default, the cal command shows the calendar of the current month. With options, we can view the calendar of the an year or particular month of ay year. This particular command will display the calendar of the year 2000.



history :-This command shows all the command used in the current login.

Chown: chow command gives us permission of ownership

**Read:** This permission allows the user to read files and in directories, it lets the user read directories and subdirectories stores in it.

**Write:** This permission allows a user to modify and delete a file. Also it allows a user to modify its contents (create, delete and rename files in it) for the directories. Unless the execute permission is not given to directories changes does do affect them.

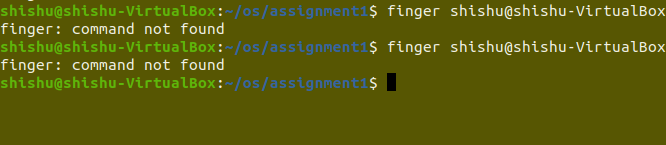
**Execute:** The write permission on a file allows it to get executed. For example, if we have a file named *php.sh* so unless we don’t give it execute permission it won’t run.

chown [OPTION]… [OWNER][:[GROUP]] FILE…

chown [OPTION]… –reference=RFILE FILE…

cal: cal command show calendar of any year which we want

**Finger** command is a user information lookup command which gives details of all the users logged in. This tool is generally used by system administrators. It provides details like login name, user name, idle time, login time, and in some cases their email address even



// sir what happening in finger command I don’t Known

Log out : using this command operating system exit from terminal



#include<stdio.h>

#include<stdlib.h>

#include<dirent.h>

#define DATA\_SIZE 1000

void createf()

{ char data[DATA\_SIZE];

char n[100];

FILE \* fPtr;

int i;

for ( i=0;i<2;i++){

printf("enter a file name:");

gets(n);

fPtr = fopen(n,"w");

if(fPtr == NULL)

{ printf("Unable to create file.\n");

exit(EXIT\_FAILURE);

}

printf("Enter contents to store in file : \n");

fgets(data, DATA\_SIZE, stdin);

fputs(data, fPtr);

fclose(fPtr);

printf("File created and saved successfully. ?? \n");

}

}

void lsandgrep(){

char fn[10],pat[10],temp[200];

FILE \*fp;

char dirname[10];

DIR\*p;

struct dirent \*d;

printf("Enter directory name\n");

scanf("%s",dirname);

p=opendir(dirname);

if(p==NULL)

{

perror("Cannot find directory");

exit(0);

}

while(d=readdir(p))

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

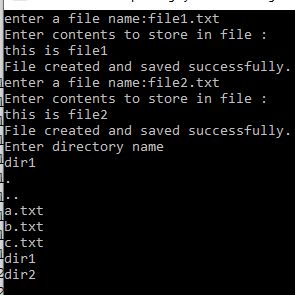
}

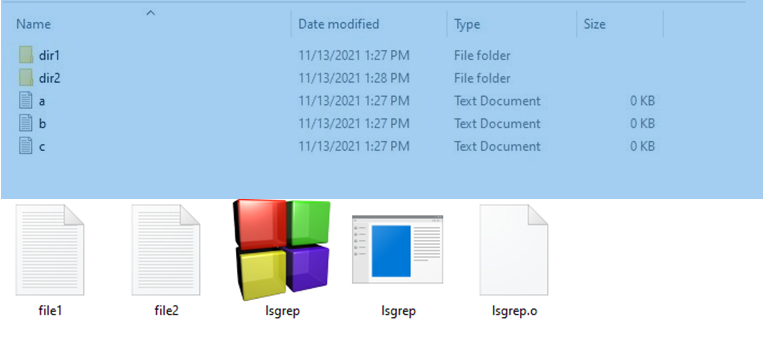
int main(){

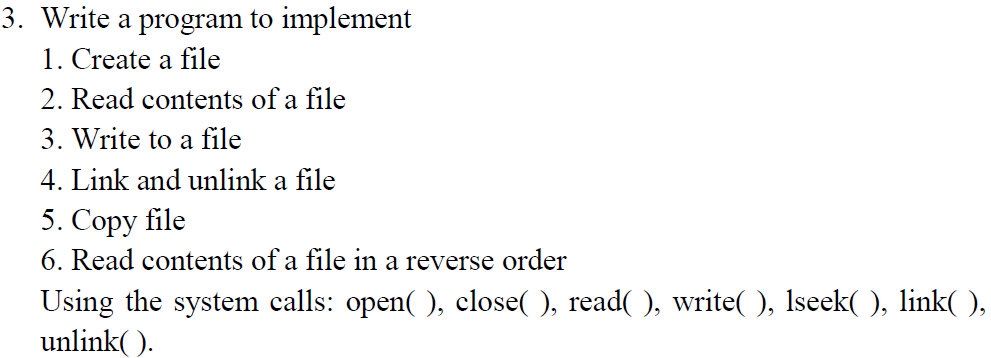
createf();

lsandgrep();

}







#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#include<dirent.h>

#define DATA\_SIZE 1000

#define MAX 100

void createf()

{ char data[DATA\_SIZE];

char n[100];

FILE \* fPtr;

printf("enter a file name:");

scanf("%s",n);

fPtr = fopen(n,"w");

if(fPtr == NULL)

{ printf("Unable to create file.\n");

exit(EXIT\_FAILURE);

}

fflush(stdin);

printf("Enter contents to store in file : \n");

fgets(data, DATA\_SIZE, stdin);

fputs(data, fPtr);

fclose(fPtr);

printf("File created and saved successfully. \n");

}

void ReadFile(){

char name[20];

printf("Enter name of file:");

scanf("%s",name);

FILE \*fp;

fp=fopen(name,"r");

if(fp==NULL)

{

printf("File does not exsist");

exit(1);

}

char ch = fgetc(fp);

while (ch != EOF)

{

printf ("%c", ch);

ch = fgetc(fp);

}

fclose(fp);

}

void CopyFile()

{

FILE \*fptr1, \*fptr2;

char filename[100], c;

printf("Enter the filename to open for reading \n");

scanf("%s", filename);

fptr1 = fopen(filename, "r");

if (fptr1 == NULL)

{

printf("Cannot open file %s \n", filename);

exit(0);

}

printf("Enter the filename to open for writing \n");

scanf("%s", filename);

fptr2 = fopen(filename, "w");

if (fptr2 == NULL)

{

printf("Cannot open file %s \n", filename);

exit(0);

}

c = fgetc(fptr1);

while (c != EOF)

{

fputc(c, fptr2);

c = fgetc(fptr1);

}

printf("\nContents copied to %s", filename);

fclose(fptr1);

fclose(fptr2);

return 0;

}

void reverseContent()

{

char x[100];

printf("Enter file name:");

scanf("%s",x);

FILE\* fp = fopen(x, "a+");

if (fp == NULL) {

printf("Unable to open file\n");

return;

}

char buf[100];

int a[MAX], s = 0, c = 0, l;

fprintf(fp, " \n");

rewind(fp);

while (!feof(fp)) {

fgets(buf, sizeof(buf), fp);

l = strlen(buf);

strcpy(a,s+l);

}

rewind(fp);

c -= 1;

while (c >= 0) {

fseek(fp, a, 0);

fgets(buf, sizeof(buf), fp);

printf("%s", buf);

c--;

}

return ;

}

int menu(){

printf("\n1.CREATE FILE\n");

printf("\n2.READ FILE\n");

printf("\n3.COPY FILE\n");

printf("\n4.REVERSE FILE\n");

printf("\n5.LINK FILE\n");

printf("\n6.UNLINK FILE\n");

printf("7.EXIT\n");

int ch;

scanf("%d",&ch);

return ch;

}

int main(){

while(1){

system("cls");

switch(menu()){

case 1:

createf();break;

case 2:

ReadFile();break;

case 3:

CopyFile();break;

case 4:

reverseContent();break;

case 5:

//linkFile();

break;

case 6:

//unlinkFile();break;

case 7:

exit(0);

break;

default:

printf("Enter valid choice!!");

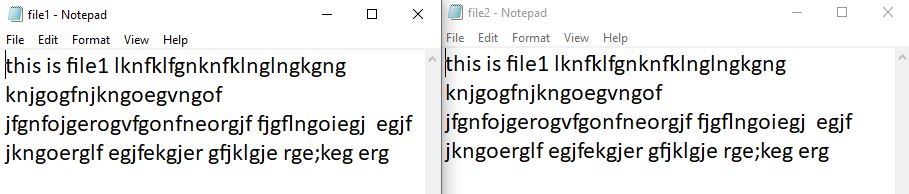
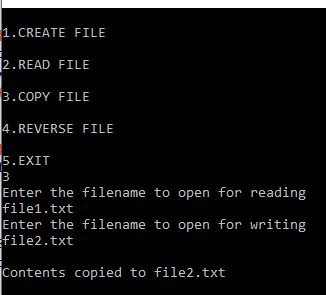
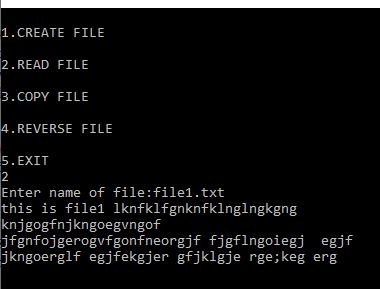
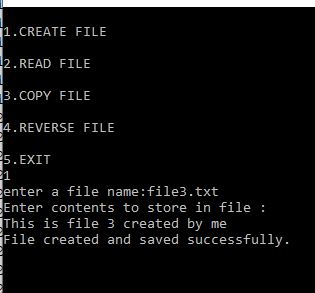
}

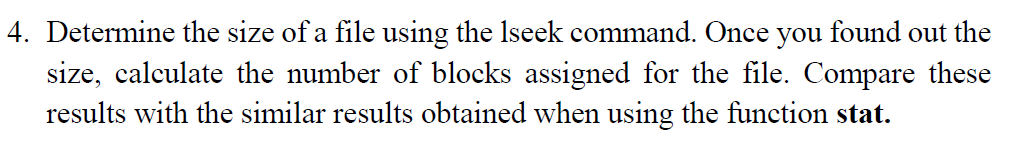
getch();

}

return 0;

}





#include <unistd.h>

#include <stdio.h>

#include <fcntl.h>

#include <sys/stat.h>

void lseekfun(int ac, char \*name[])

{

if ( ac < 2 ) return 0;

int fd = open(name,O\_RDONLY);

int size = lseek(fd, 0, SEEK\_END);

printf("Size using lseek:%d\n", size);

close(fd);

return 0;

}

void statfun(int ac, char \*name[])

{

if ( ac < 2 ) return 0;

struct stat stbuf;

stat( name, &stbuf);

printf ("Size using stat:%lld\n", stbuf.st\_size);

}

int main(){

printf("Enter file name:");

char name[20];

scanf("%s",name);

int n=strlen(name);

lseekfun(n,name);

statfun(n,name);

return 0;

}

