***File Api assignment***

**1. Program to retrieve the permission of a file**

#include <stdio.h>

#include <stdlib.h>

#include <sys/stat.h>

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

{

    char \*filename;

    struct stat fs;

    int r;

    if( argc<2 )

    {

        puts("Filename required");

        exit(1);

    }

    filename = argv[1];

    printf("Obtaining permission mode for '%s':\n",filename);

    r = stat(filename,&fs);

    if( r==-1 )

    {

        fprintf(stderr,"File error\n");

        exit(1);

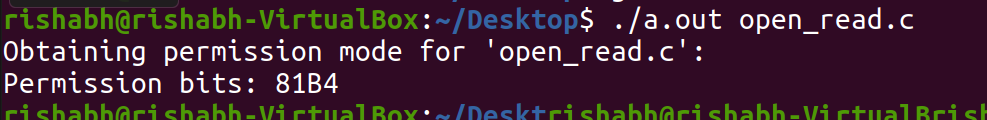
    }

    /\* file permissions are kept in the st\_mode member \*/

    printf("Permission bits: %X\n",fs.st\_mode);

    return(0);

}



#include <stdio.h>

#include <unistd.h>

#include <sys/stat.h>

#include <time.h>

void printFileProperties(struct stat stats);

int main()

{

    char path[100];

    struct stat stats;

    printf("Enter source file path: ");

    scanf("%s", path);

    // stat() returns 0 on successful operation,

    // otherwise returns -1 if unable to get file properties.

    if (stat(path, &stats) == 0)

    {

        printFileProperties(stats);

    }

    else

    {

        printf("Unable to get file properties.\n");

        printf("Please check whether '%s' file exists.\n", path);

    }

    return 0;

}

void printFileProperties(struct stat stats)

{

    // File permissions

    printf("\nFile access: ");

    if (stats.st\_mode & R\_OK)

        printf("read ");

    if (stats.st\_mode & W\_OK)

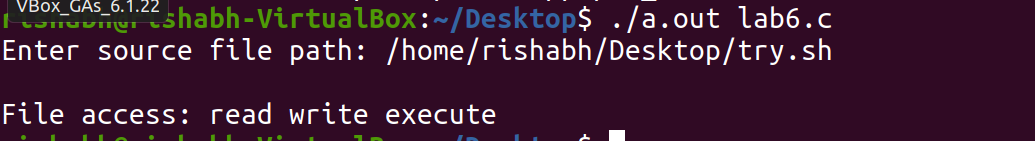
        printf("write ");

    if (stats.st\_mode & X\_OK)

        printf("execute");

     printf("\n");

}



**2.Program to find the number of links**  
#include <stdio.h>

#include <sys/stat.h>

int main ( int argc, char \*\* argv ) {

    int i;

    struct stat st;     /\* stat puts info here \*/

    for (i = 1; i < argc; ++i) {

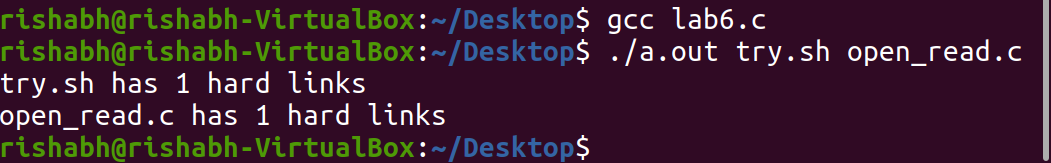
        if (stat(argv[i], &st) == -1) perror(argv[i]);

        else printf("%s has %ld hard links\n", argv[i], st.st\_nlink);

    }

    return 0;

}



**3.Program to find the user and group name**  
#include <pwd.h>

#include <grp.h>

#include <sys/stat.h>

#include <stdio.h>

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

{

struct stat info;

for(int i=1;i<argc;i++){

stat(argv[1],&info);

struct passwd \*pw = getpwuid(info.st\_uid);

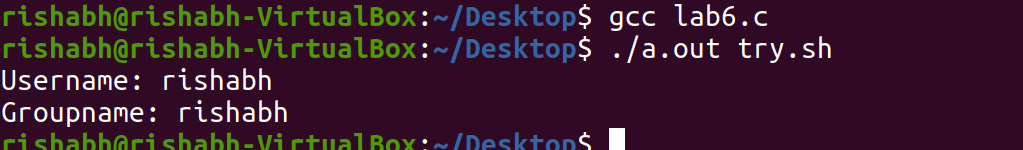
struct group  \*gr = getgrgid(info.st\_gid);

printf("Username: %s\n",pw->pw\_name);

printf("Groupname: %s \n",gr->gr\_name);

}

}



**4.Program to find the type of the file. The input will be given as command line argument.**

#include <stdio.h>

#include <stdlib.h>

#include <sys/stat.h>

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

{

const char \*filename;

     struct stat fs;

     int st;

     if( argc<2 )

     {

         puts("Filename is not entered! Exit(0)..");

         exit(1);

     }

     filename = argv[1];

     st = stat(filename,&fs);

     if( st==-1 )

     {

         fprintf(stderr,"File error!\n");

         exit(1);

     }

     if( S\_ISREG(fs.st\_mode) )

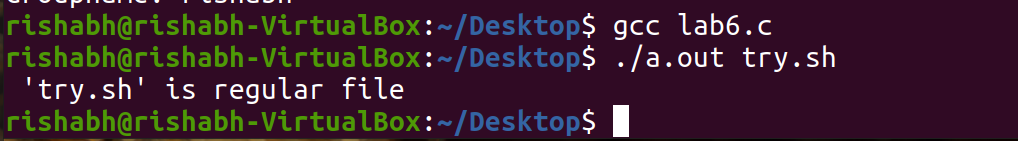
         printf(" '%s' is regular file\n",filename);

     else

         printf(" '%s' is not a regular file\n",filename);

     return(0);

}

****

**5.Program to create a file f1.txt. Write to the file “We are quarantined since one month  
“ and the move the file pointer to 50 bytes ahead and then write “we are not sure when lock down will be released”**

#include<stdio.h>

#include<unistd.h>

#include<sys/types.h>

#include<iostream>

#include<fcntl.h>

 using namespace std;

int main()

{  int fd1;

 char b1[]="We are quarantined since one month";

char b2[]="we are not sure when lock down will be released";

fd1=creat("covid.txt",0622);

if(fd1<0)

{

cout<<"Error in creating file"<<endl;

  return 0;  }

write(fd1,b1,35);

lseek(fd1,50,SEEK\_SET);

write(fd1,b2,50);

close(fd1);

 return 0;

}

**6.Program to read the first 10 bytes of data from the file f1.txt and again read the file contents from 20 th byte.**

#include<unistd.h>

#include<fcntl.h>

#include<sys/types.h>

#include<stdio.h>

#include<sys/stat.h>

int main()

{

int n,f;

char buff[100];

f=open("commands.txt",O\_RDWR);

read(f,buff,10);

write(1,buff,10);

lseek(f,20,SEEK\_SET);

read(f,buff,85);

write(1,buff,85);

}

