



Escuela
Superior
de Cómputo



INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO

Práctica No.2

Introducción a los sistemas operativos Linux y
Windows (2)

Unidad de aprendizaje: Sistemas Operativos

Grupo: 2CM8

Integrantes del equipo:

Domínguez Morán Joaquín
Carrillo Balcazar Eduardo Yair
Ruiz López Luis Carlos

Profesor:

Jorge Cortes Galicia

17 de septiembre de 2018

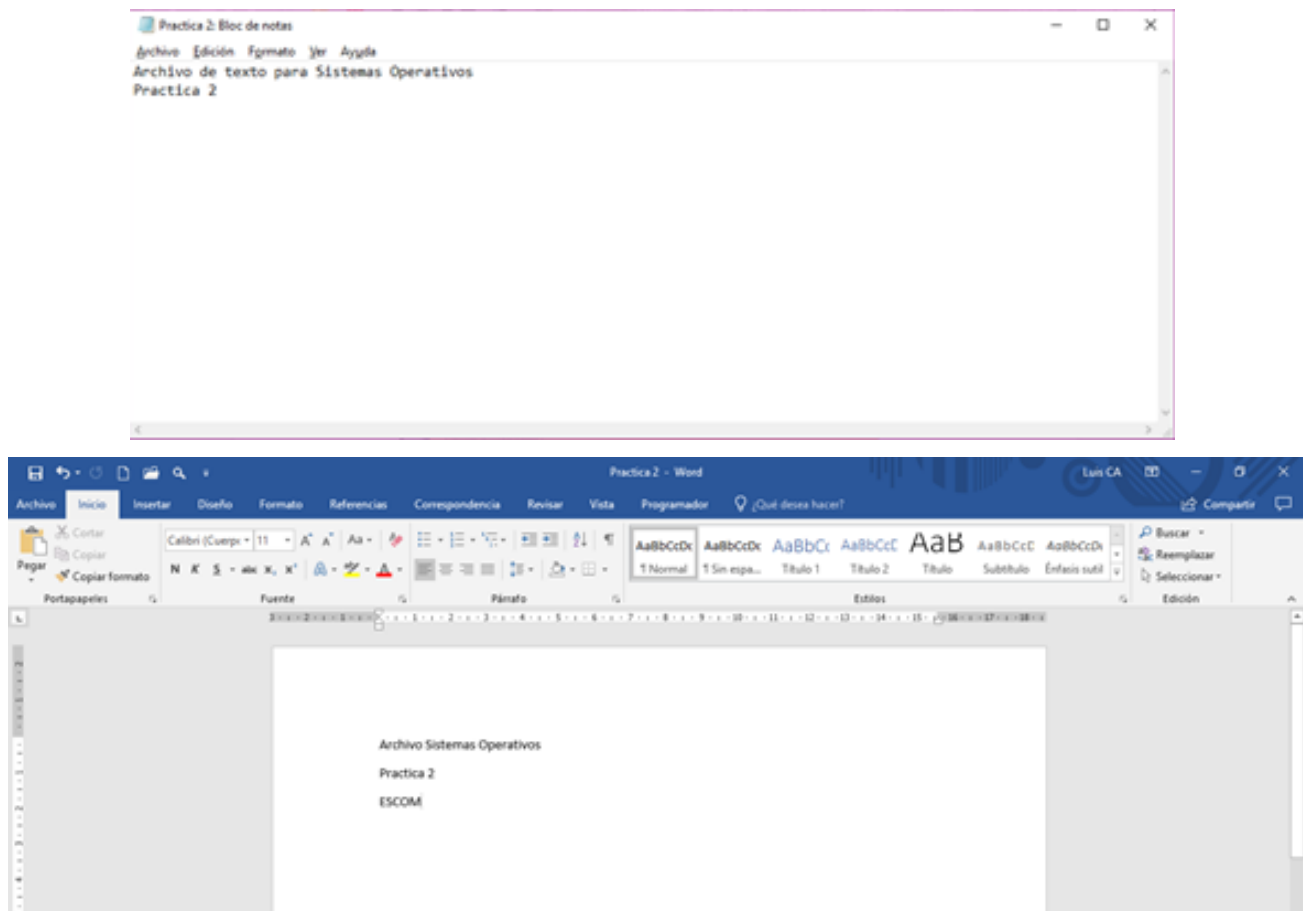
1. Competencias.

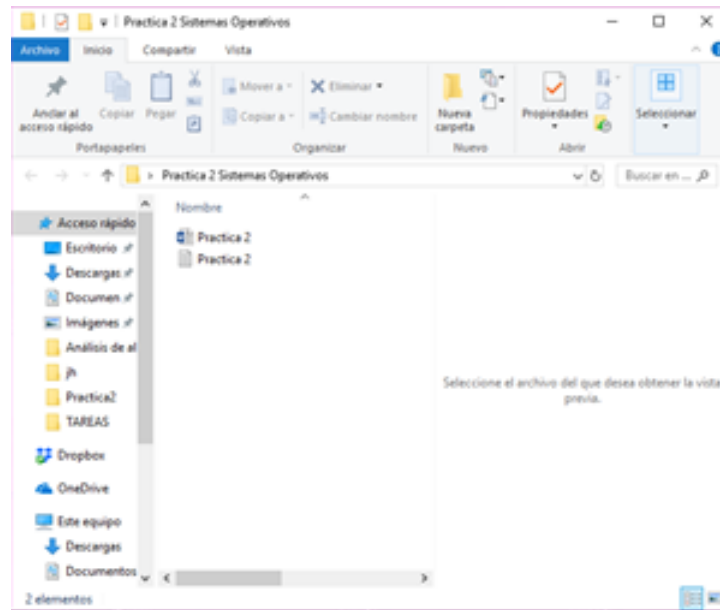
El alumno aprende a familiarizarse con los sistemas operativos Linux y Windows mediante el uso de la interfaz de llamadas al sistema respectiva de cada sistema operativo, a través del desarrollo de programas bajo lenguaje C para la invocación de llamadas al sistema propias de los sistemas operativos revisados.

2. Desarrollo.

Diferencias entre archivos de texto.

1.-Se crean los archivos de texto en Windows:



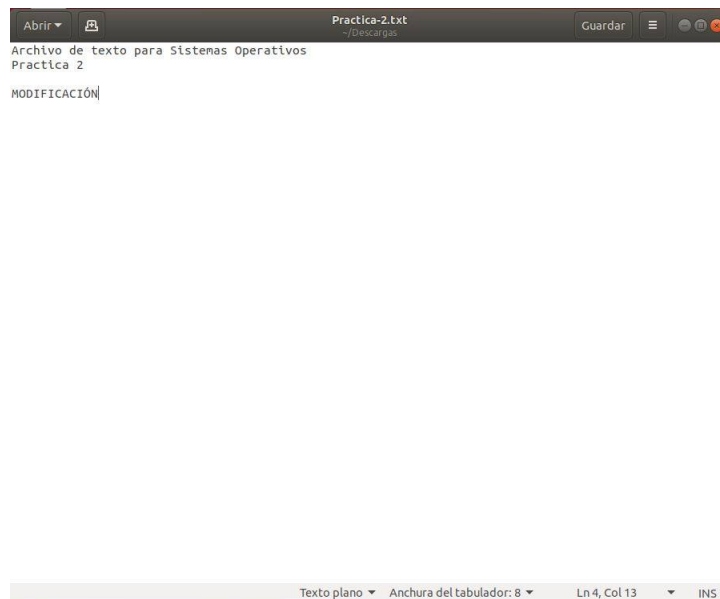


2.-Se inicia sesion en Linux

3.-Se monta la memoria USB

4.-Editamos los archivos de texto txt y word:

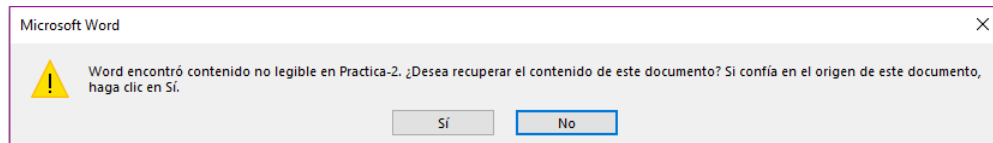
Imágenes/Documentos/41804851_314050915814812_298327





5.-Abrimos los archivos de texto en Windows





El archivo txt se ve bien, con todas las modificaciones que realizamos en Linux. Por otro lado, el archivo word no se pudo abrir; esto se debe a la diferencia de caracteres con los que trabajan.

2.1. Llamadas al sistema en Linux.

A continuación se explican las principales llamas del sistena Linux.

Llamada al sistema.	Descripción.
open	La llamada al sistema open () abre el archivo especificado por la ruta de acceso. Si el el archivo especificado no existe, puede opcionalmente ser creado por open(). El valor de retorno de open () es un descriptor de archivo, un número entero, no negativo, pequeño que se usa en otras llamadas al sistema para referirse al archivo abierto.
close	Cambiar bits a modo de archivo.
read	Puede ser usado para editar todo tipo de texto sin formato. Es especialmente útil para editar programas.
write	La función write () intentará escribir n bytes desde el buffer señalado al archivo asociado con el archivo abierto.
creat	Crea un nuevo archivo o reescribe uno existente.
lseek	Mover el desplazamiento de archivo de lectura / escritura.
access	Comprueba si el proceso de llamada puede acceder al nombre de ruta del archivo. Si el nombre de ruta del archivo es un enlace simbólico, se desreferencia.
stat	Devuelve información sobre un archivo. No se requieren permisos en el archivo en sí, pero en el caso de stat se requiere permiso de ejecución (búsqueda) en todos los directorios en la ruta que conducen al archivo.
chmod	Cambia los permisos de un archivo
chown	Cambia de propietario y grupo de archivos.
fcntl	Manipula el descriptor de archivo.
chdir	Cambia el directorio de trabajo.
mkdir	Crea un directorio.
opendir	Abrirá una secuencia de directorio correspondiente al directorio nombrado por el argumento
readdir	Leer un directorio.

A continuación se explican las principales llamas del sistema Windows.

Llamada al sistema.	Descripción.
OpenFile	Crea, abre, vuelve a abrir o elimina un archivo. Esta función tiene capacidades limitadas y no es recomendable.
CloseFile (CloseHandle)	Cierra un identificador de objeto abierto. Si un archivo está abierto cuando una aplicación finaliza, el sistema lo cierra automáticamente.
ReadFile	Lee datos del archivo o dispositivo de entrada / salida (E / S) especificado. Las lecturas se producen en la posición especificada por el puntero del archivo si el dispositivo lo admite. Esta función está diseñada para operaciones sincrónicas y asíncronas.
WriteFile	Escribe datos en el archivo o dispositivo de entrada / salida (E / S) especificado. Esta función está diseñada para operación síncrona y asíncrona.
CreateFile	Crea o abre un archivo o dispositivo de E / S. Los dispositivos de E / S utilizados más comúnmente son los siguientes: archivo, secuencia de archivos, directorio, disco físico, volumen, búfer de consola, unidad de cinta, recurso de comunicaciones, ranura de correo y tubería. La función devuelve un identificador que se puede utilizar para acceder al archivo o dispositivo para varios tipos de E / S según el archivo o dispositivo y las banderas y atributos especificados.
SetFilePointer	Mueve el puntero de archivo del archivo especificado. Esta función almacena el puntero del archivo en dos valores LONG.
CreateDirectory	Crea un nuevo directorio Si el sistema de archivos subyacente admite seguridad en archivos y directorios, la función aplica un descriptor de seguridad específico al nuevo directorio.
SetCurrentDirectory	Cambia el directorio actual para el proceso actual.

A continuación se explican las principales funciones en Windows.

Función.	Descripción.
stat	Obtiene el estado del archivo. La función stat () obtendrá información sobre el archivo nombrado. El argumento de ruta apunta a un nombre de ruta nombrando un archivo. No se requiere leer, escribir o ejecutar el permiso del archivo nombrado.
opendir	Abre un directorio. La función opendir () abrirá una secuencia de directorios correspondiente al directorio nombrado.
readdir	Lee un directorio. La función readdir () devolverá un puntero a una estructura que representa la entrada de directorio en la posición actual en la secuencia de directorio especificada por el argumento de dirección del directorio , y posicionará la secuencia de directorio en la entrada siguiente. Devolverá un puntero nulo al llegar al final de la secuencia de directorio.

La razón por la que no existe una llamada al sistema similar a chmod de Linux, la cual modifica los permisos de un archivo, está dada por el Sistema de Archivos que maneja cada sistema operativo.

Los sistemas UNIX o compatibles POSIX, incluyendo sistemas basados en Linux y Mac OS X, poseen un sistema simple para el manejo de permisos sobre archivos individuales. POSIX especifica también un sistema de listas de control de acceso (ACLs), pero sólo está implementado por ciertos sistemas de archivos y sistemas operativos.

Las variantes de DOS (incluyendo los productos de Microsoft MS-DOS, Windows 95, Windows 98, y Windows Me) no implementan ningún sistema de permisos. Existe un atributo de "solo lectura" un atributo de "archivo oculto" que pueden ser asignados o quitados de cualquier archivo por cualquier usuario.

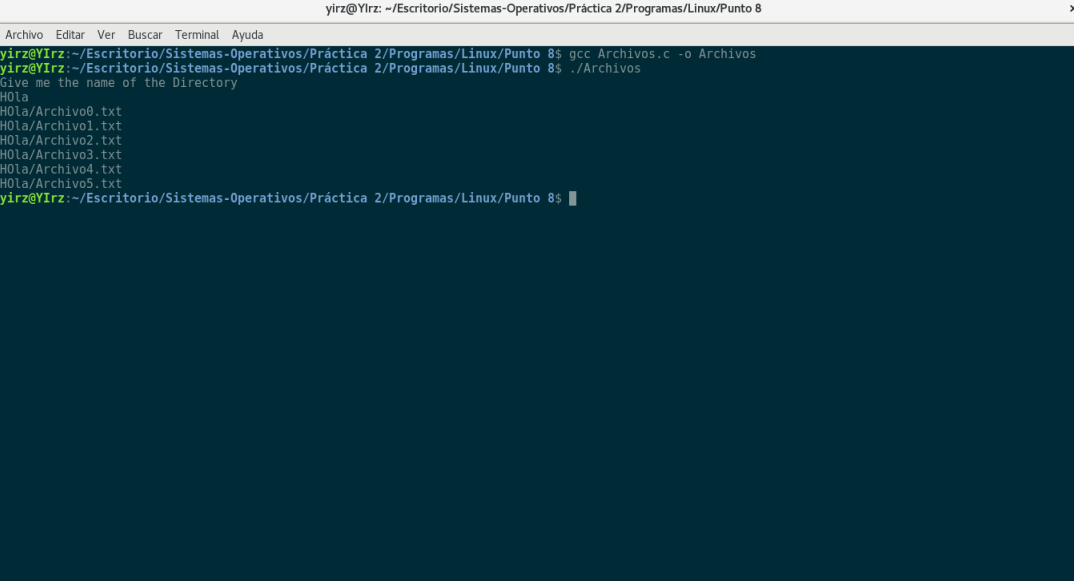
Microsoft Windows NT y sus derivados (incluyendo Windows 2000 y Windows XP), así como VMS y OpenVMS usan listas de control de acceso (ACLs) para administrar un conjunto más complejo y variado de permisos. No tienen un atributo específico en el sistema de archivos que garantice su respectivo permiso de ejecución.

2.2. Programas Desarrollados

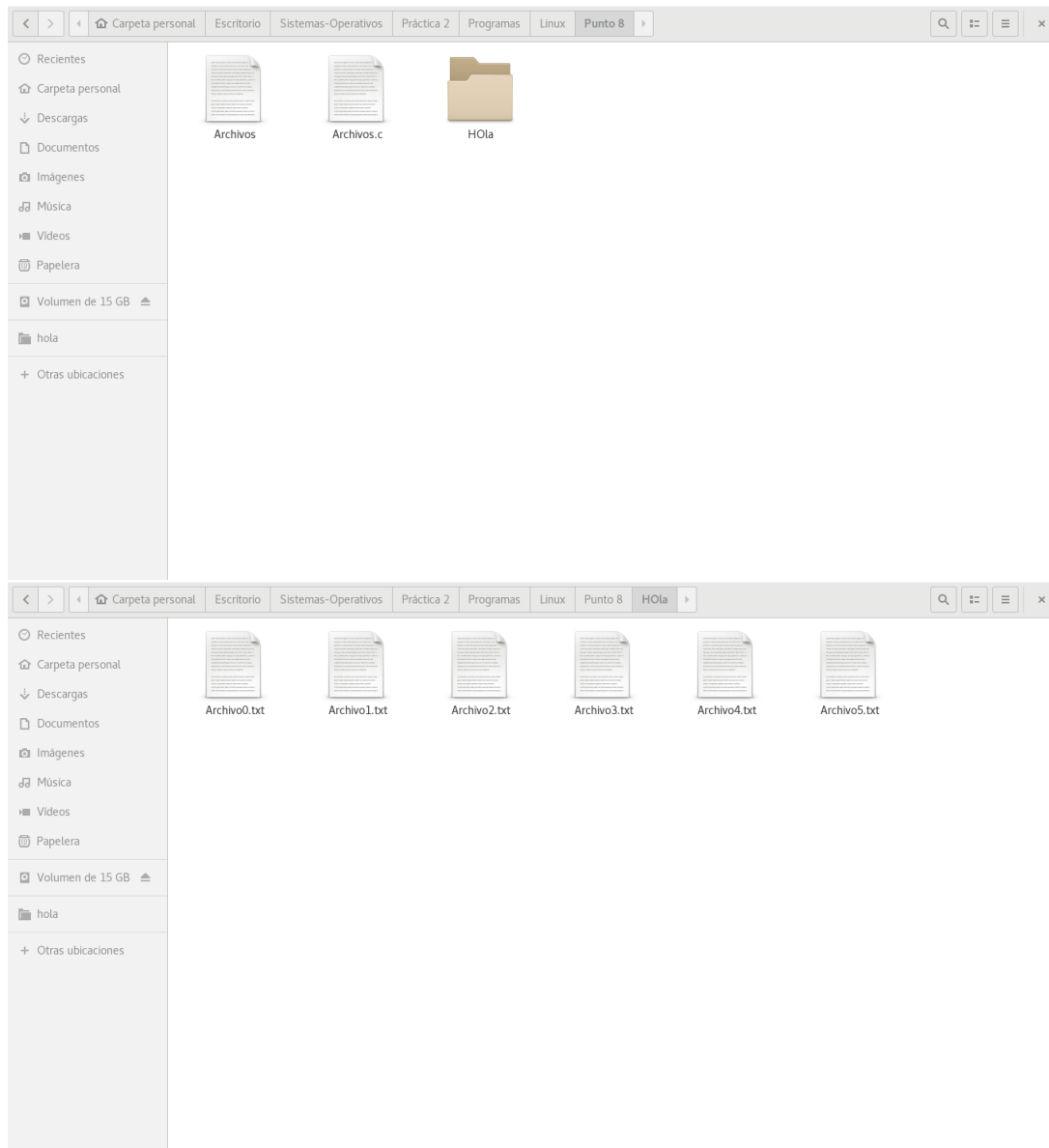
1. Utilizando las llamadas al sistema revisadas para Linux que sean necesarias, desarrolle un programa en C que cree una serie aleatoria de archivos (en una ruta especificada a través de una línea de comando, el directorio no debe existir previamente), el contenido de los archivos serán cadenas que estén almacenadas en un arreglo.
2. Una vez creados los archivos con sus contenidos por el programa del punto 8 y utilizando las llamadas al sistema revisadas para Linux que sean necesarias, desarrolle un programa en C para cambiar los permisos de un archivo seleccionado por el usuario.
3. Una vez creados los archivos con sus contenidos por el programa del punto 8 y utilizando las llamadas al sistema revisadas para Linux que sean necesarias, desarrolle un programa en C que liste los archivos, creados mostrando su tamaño, fecha y hora de acceso.
4. Una vez creados los archivos con sus contenidos por el programa del punto 8 y utilizando las llamadas al sistema revisadas para Linux que sean necesarias, desarrolle un programa en C para mostrar el contenido de un archivo seleccionado por el usuario, y que copie uno o más de los archivos creados a un directorio previamente establecido.
5. Desarrolle las versiones para Windows de los programas descritos en los puntos 1, 3 y 11, utilizando las llamadas al sistema revisadas para Windows que sean necesarias.

2.2.1. Linux

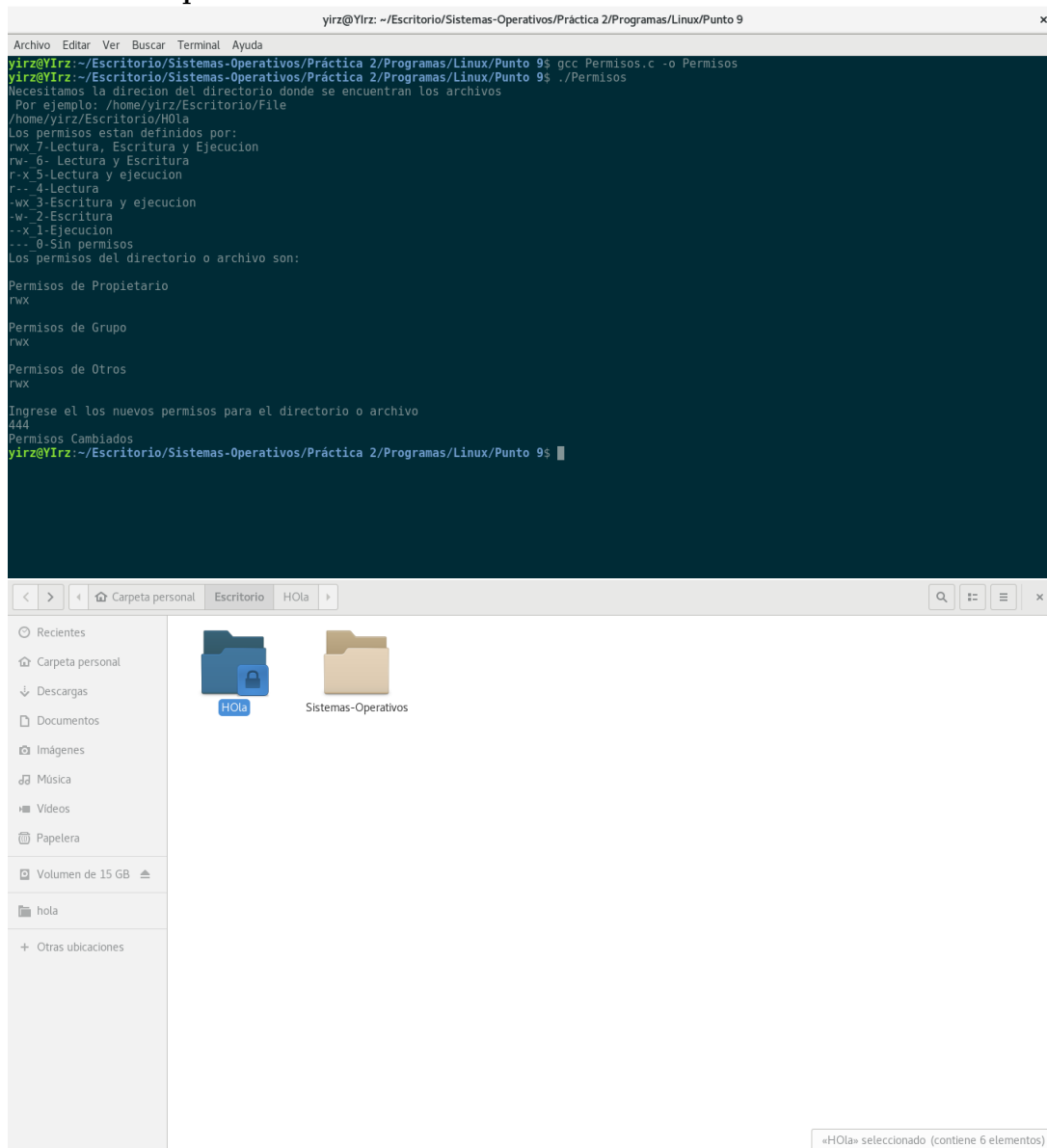
■ Creación de Archivos



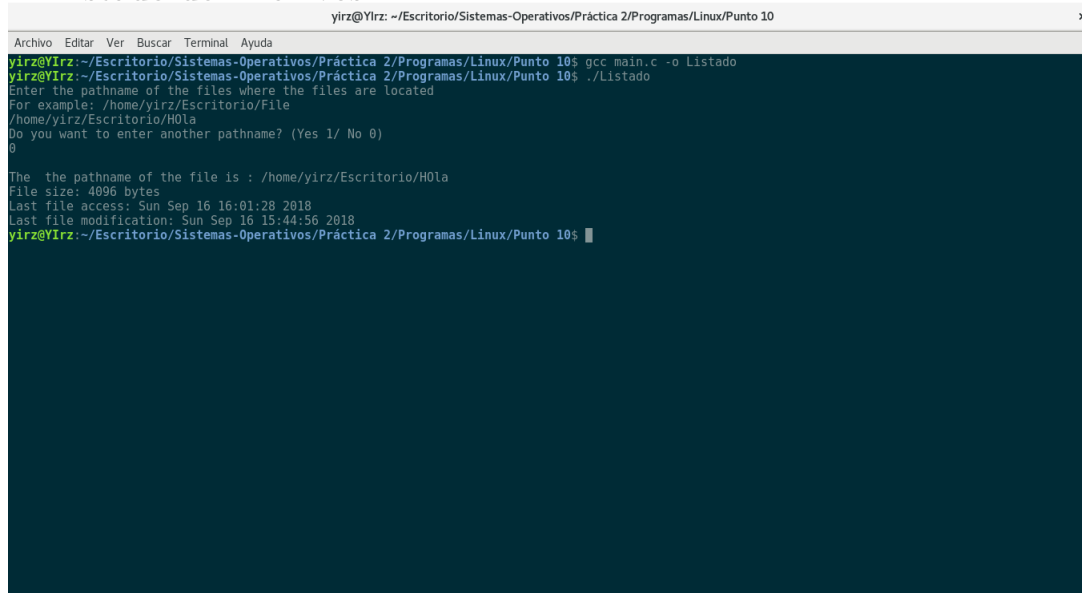
```
yirz@Yirz: ~/Escritorio/Sistemas-Operativos/Práctica 2/Programas/Linux/Punto 8
Archivo Editar Ver Buscar Terminal Ayuda
yirz@Yirz:~/Escritorio/Sistemas-Operativos/Práctica 2/Programas/Linux/Punto 8$ gcc Archivos.c -o Archivos
yirz@Yirz:~/Escritorio/Sistemas-Operativos/Práctica 2/Programas/Linux/Punto 8$ ./Archivos
Give me the name of the Directory
H01a
H01a/Archivo0.txt
H01a/Archivo1.txt
H01a/Archivo2.txt
H01a/Archivo3.txt
H01a/Archivo4.txt
H01a/Archivo5.txt
yirz@Yirz:~/Escritorio/Sistemas-Operativos/Práctica 2/Programas/Linux/Punto 8$
```



■ Cambio de permisos



- **Enlistado de Archivos**



```
yirz@Yirz: ~/Escritorio/Sistemas-Operativos/Práctica 2/Programas/Linux/Punto 10
Archivo Editar Ver Buscar Terminal Ayuda
yirz@Yirz:~/Escritorio/Sistemas-Operativos/Práctica 2/Programas/Linux/Punto 10$ gcc main.c -o Listado
yirz@Yirz:~/Escritorio/Sistemas-Operativos/Práctica 2/Programas/Linux/Punto 10$ ./Listado
Enter the pathname of the files where the files are located
For example: /home/yirz/Escritorio/File
/home/yirz/Escritorio/H0la
Do you want to enter another pathname? (Yes 1/ No 0)
0
The the pathname of the file is : /home/yirz/Escritorio/H0la
File size: 4096 bytes
Last file access: Sun Sep 16 16:01:28 2018
Last file modification: Sun Sep 16 15:44:56 2018
yirz@Yirz:~/Escritorio/Sistemas-Operativos/Práctica 2/Programas/Linux/Punto 10$
```

- **Mostrar contenido de un archivo y copiado de archivos**



```
Escriba cual es el archivo que desea abrir para ver su contenido, recuerde escri
bir la extension del mismo: escom.txt

voca 3 :3 <3
Ok No

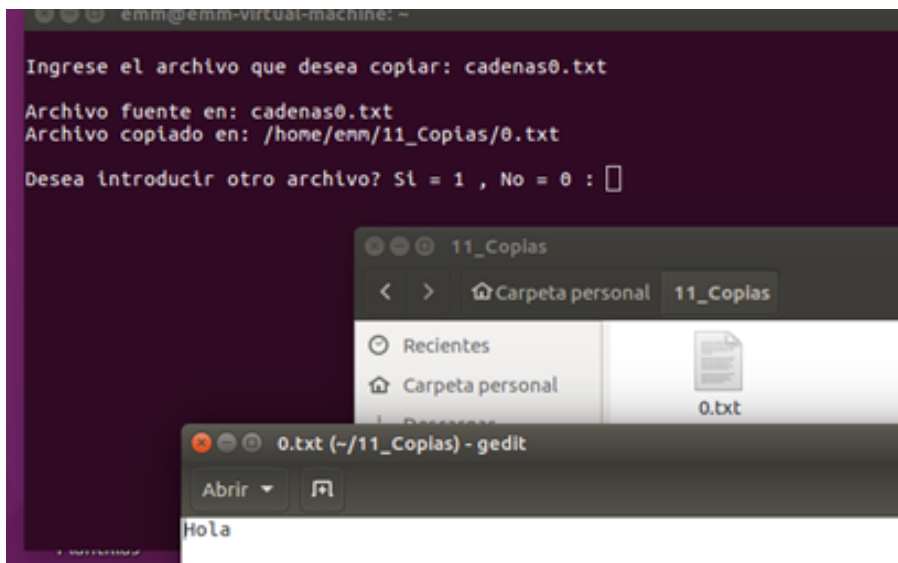
Que desea hacer?
1. Mostrar contenido de un archivo
2. Copiar archivo al directorio previamente establecido
3. Salir

```

escom.txt (~/) - gedit

Abrir

```
voca 3 :3 <3
Ok No
```



2.2.2. Windows

■ Creación de Archivos

The screenshot shows a Windows command prompt window and a File Explorer window. The command prompt window displays the following commands and output:

```

C:\Windows\System32\cmd.exe
Microsoft Windows [Versión 10.0.17134.228]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Yirz\Documents\Yirz\5to Semestre\Sistemas-Operativos\Práctica 2\Programas\Windows\Punto 8>gcc Archivos.c -o Archivos

C:\Users\Yirz\Documents\Yirz\5to Semestre\Sistemas-Operativos\Práctica 2\Programas\Windows\Punto 8>Archivos
Give me the name of the Directory
HOLA
HOLA/Archivo0.txt
HOLA/Archivo1.txt
HOLA/Archivo2.txt
HOLA/Archivo3.txt
HOLA/Archivo4.txt
HOLA/Archivo5.txt
HOLA/Archivo6.txt
HOLA/Archivo7.txt

C:\Users\Yirz\Documents\Yirz\5to Semestre\Sistemas-Operativos\Práctica 2\Programas\Windows\Punto 8>

```

The File Explorer window shows the directory structure: Este equipo > Documentos > Yirz > 5to Semestre > Sistemas-Operativos > Práctica 2 > Programas > Windows > Punto 8. The 'HOLA' folder is selected, and the contents are listed in a table below.

Nombre	Fecha de modifica...	Tipo	Tamaño
Archivo0	16/09/2018 16:51	Documento de tex	1 KB
Archivo1	16/09/2018 16:51	Documento de tex	1 KB
Archivo2	16/09/2018 16:51	Documento de tex	1 KB
Archivo3	16/09/2018 16:51	Documento de tex	1 KB
Archivo4	16/09/2018 16:51	Documento de tex	1 KB
Archivo5	16/09/2018 16:51	Documento de tex	1 KB
Archivo6	16/09/2018 16:51	Documento de tex	1 KB
Archivo7	16/09/2018 16:51	Documento de tex	1 KB

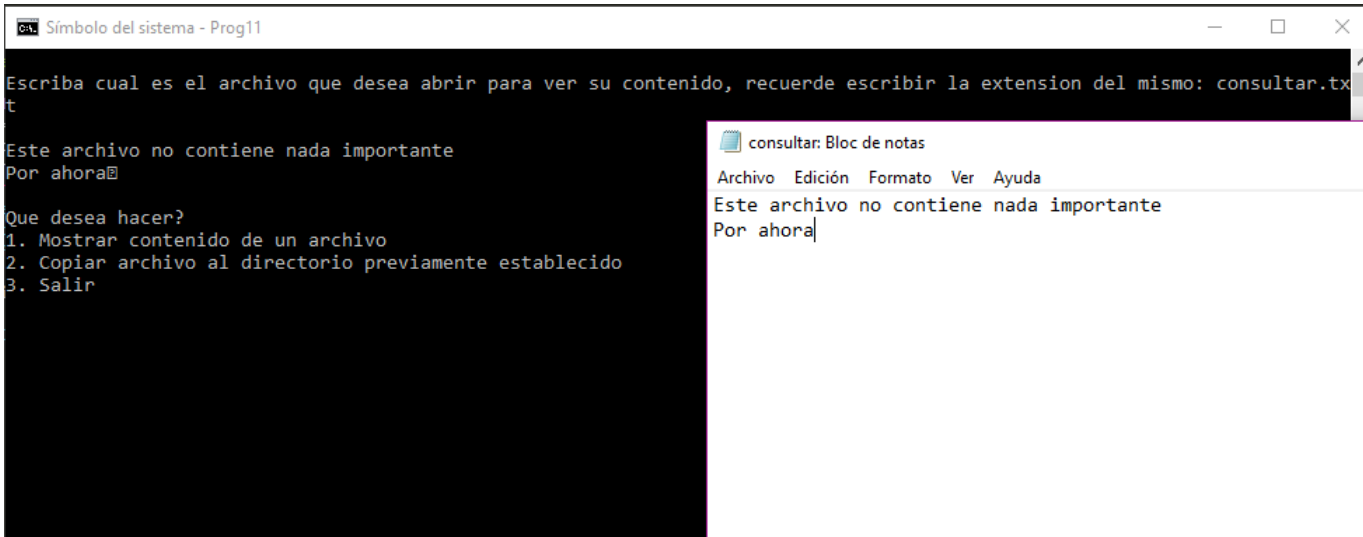
■ Enlistado de Archivos

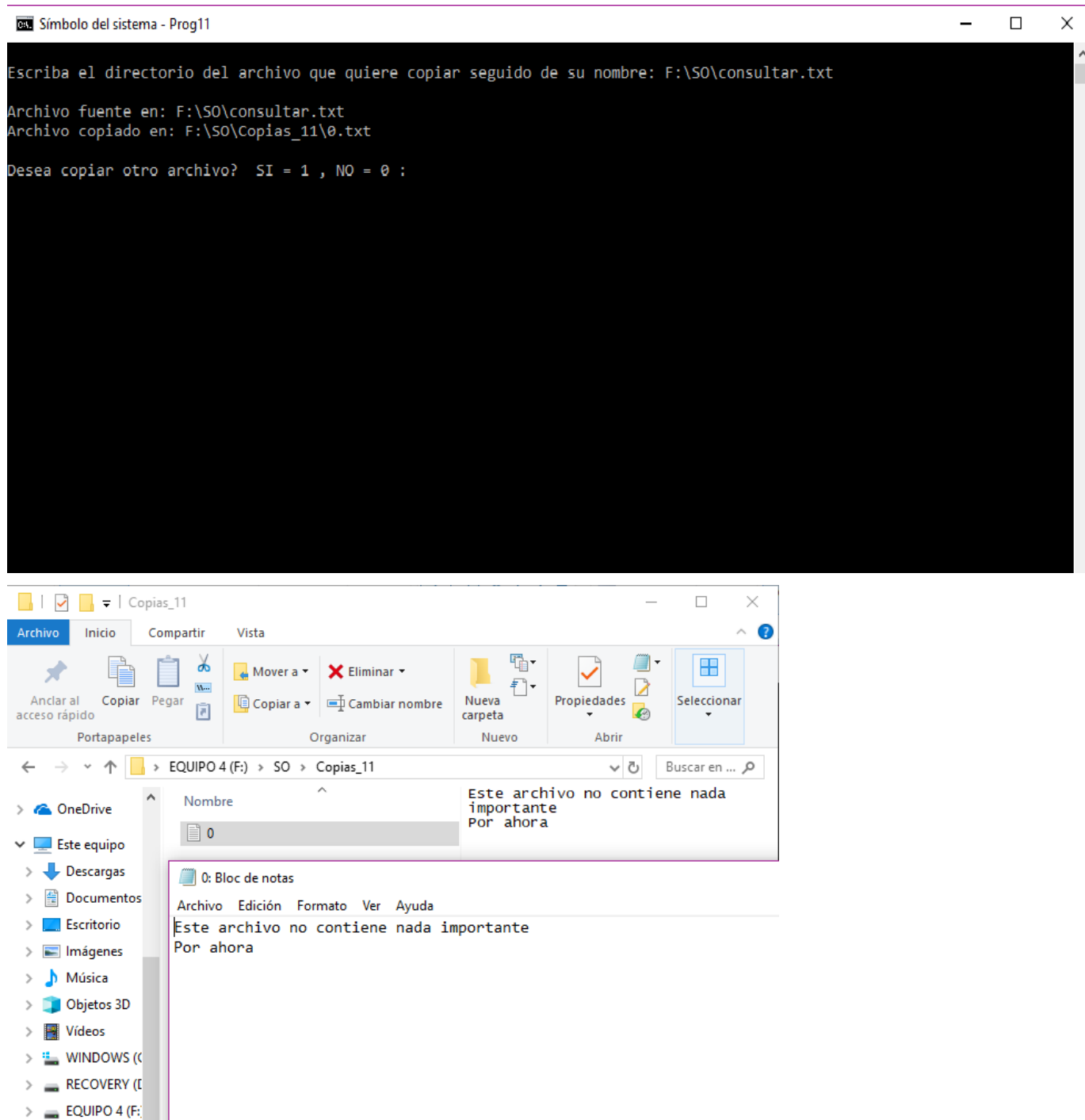
C:\Windows\System32\cmd.exe - Listado

```
C:\Users\Yinz\Documents\Yinz\5to Semestre\Sistemas-Operativos\Práctica 2\Programas\Windows\Punto 10>gcc Listado.c -o Listado
C:\Users\Yinz\Documents\Yinz\5to Semestre\Sistemas-Operativos\Práctica 2\Programas\Windows\Punto 10>Listado
Ingresa el directorio:C:\Users\Yinz\Desktop\HOLA
Archivo0.txt      TAMANIO:   -5238978952695414784  Hora de acceso: 09/16/18 - 05:03PM
Archivo1.txt      TAMANIO:  18085037836599296  Hora de acceso: 09/16/18 - 05:03PM
Archivo2.txt      TAMANIO:  18085037836599296  Hora de acceso: 09/16/18 - 05:03PM
Archivo3.txt      TAMANIO:  18085037836599296  Hora de acceso: 09/16/18 - 05:03PM
Archivo4.txt      TAMANIO:  18085037836599296  Hora de acceso: 09/16/18 - 05:03PM
Archivo5.txt      TAMANIO:  18085037836599296  Hora de acceso: 09/16/18 - 05:03PM
Archivo6.txt      TAMANIO:  18085037836599296  Hora de acceso: 09/16/18 - 05:03PM
Archivo7.txt      TAMANIO:  18085037836599296  Hora de acceso: 09/16/18 - 05:03PM
Presione una tecla para continuar . . .
```

■ Mostrar contenido de un archivo y copiado de archivos

```
Que desea hacer?
1. Mostrar contenido de un archivo
2. Copiar archivo al directorio previamente establecido
3. Salir
```





2.3. Código Fuente.

2.3.1. Linux

Creación de Archivos

```
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <sys/stat.h> //INCLUYE EL MKDIR
#include <fcntl.h>
#include <time.h>
#include <stdlib.h>

int main(int argc, char const *argv[])
{
    char name[]="Archivo .txt";
    char content[]="Hola Mundo";
    char dir[100];
    char aux[100];
    int NFile,IdFile;
    printf("Give me the name of the Directory\n");
    scanf("%s",dir);
    if(mkdir(dir,0777)==-1){
        printf("Directory don't create\n");
    }
    else{
        NFile=rand()%11;

        for (int i = 0; i < NFile; ++i)
        {
            strcpy(aux,"./");
            strcpy(aux,dir);
            name[7]=i+'0';
            strcat(aux,"/");
            strcat(aux,name);
            printf("%s\n", aux);
            creat(aux,0777);
            IdFile=open(aux,O_RDWR);
            write(IdFile,content,sizeof(content));
            strcpy(aux,"\\0");

        }
    }
    return 0;
}
```

Cambio de permisos

```

#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <time.h>
#include <stdlib.h>

int main(){
    char Permission[10];
    int Permission_Octal;
    char direction[200];
    struct stat file; // para apuntar al archivo que deseamos cambiar los permisos
    printf("Necesitamos la direccion del directorio donde se encuentran los archivos\n");
    printf("Por ejemplo: /home/yirz/Escritorio/File\n");
    scanf("%s",direction);
    if(stat(direction,&file)==0){
        printf("Los permisos estan definidos por:\n");
        printf("rwx_7-Lectura, Escritura y Ejecucion\n");
        printf("rw-_6- Lectura y Escritura\n");
        printf("r-x_5-Lectura y ejecucion\n");
        printf("r--_4-Lectura\n");
        printf("-wx_3-Escritura y ejecucion\n");
        printf("-w-_2-Escritura\n");
        printf("--x_1-Ejecucion\n");
        printf("---_0-Sin permisos\n");
        printf("Los permisos del directorio o archivo son:\n");
        printf("\n");
        //Propietario
        printf("Permisos de Propietario");
        printf( (file.st_mode & S_IRUSR) ? "r" : "-");
        printf( (file.st_mode & S_IWUSR) ? "w" : "-");
        printf( (file.st_mode & S_IXUSR) ? "x" : "-");
        printf("\n\n");
        //GRUPO
        printf("Permisos de Grupo\n");
        printf( (file.st_mode & S_IRGRP) ? "r" : "-");
        printf( (file.st_mode & S_IWGRP) ? "w" : "-");
        printf( (file.st_mode & S_IXGRP) ? "x" : "-");
        printf("\n\n");
        //OTROS
        printf("Permisos de Otros\n");
        printf( (file.st_mode & S_IROTH) ? "r" : "-");
        printf( (file.st_mode & S_IWOTH) ? "w" : "-");
        printf( (file.st_mode & S_IXOTH) ? "x" : "-");
        printf("\n\n");
        printf("Ingrese el los nuevos permisos para el directorio o archivo\n");
        scanf("%s",Permission);
        Permission_Octal=strtol(Permission,0,8);
        if(chmod(direction,Permission_Octal)==0)
            printf("Permisos Cambiados\n");
        else
            printf("Permisos no cambiados\n");
    }
}

```

```

    }
    else{
        printf("Dritectorio o archivo no existente\n");
    }
    return 0;
}

```

Enlistado de Archivos

```

#include <sys/types.h>
#include <sys/stat.h>
#include <time.h>
#include <stdio.h>
#include <stdlib.h>
#define SIZE1 10
#define SIZE2 100
#define TRUE 1
#define FALSE 0

int getPath(char paths[][SIZE2])
{
    int i=0;
    int condition=TRUE;
    do{
        printf("Enter the pathname of the files where the files are located\n");
        printf("For example: /home/yirz/Escritorio/File\n");
        scanf("%s",&paths[i][SIZE2]);
        i++;
        printf("Do you want to enter another pathname? (Yes 1/ No 0)\n");
        scanf (" %d", &condition);
    }while( condition==TRUE);
    return i;
}

int main(int argc, char const *argv[])
{
    char paths[SIZE1][SIZE2];
    struct stat buffer;
    int status;
    int i,j;
    i=getPath(paths);

    for (j = 0; j <= i; j++)
    {
        status = stat(&paths[j][0], &buffer);
        if(status==0)
        {
            printf("\nThe the pathname of the file is : %s\n", &paths[j][0]);
            printf("File size: %lld bytes\n",(long long) buffer.st_size);
            printf("Last file access: %s", ctime(&buffer.st_atime));
            printf("Last file modification: %s", ctime(&buffer.st_mtime));
        }
    }
    return 0;
}

```

Mostrar contenido de un archivo y copiado de archivos

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

int Mostrar()
{
    char archi[50];
    int fd=0;
    char c;
    printf("\nEscriba cual es el archivo que desea abrir para ver su contenido, recuerde el nombre\n");
    scanf("%s", archi);

    fd = open(archi,O_RDONLY);
    printf("\n");

    if(fd!=-1) //Si el archivo existe...
    {
        //... se lee caracter por caracter y cada uno de estos es mostrado en pantalla.
        while(read(fd,&c,sizeof(c))!=0){
            printf("%c",c);
        }
        //Se cierra el archivo y la opcion del menu .
        close(fd);
    }

    else
        printf("\nEl archivo no existe\n");

    return 0;
}

int Copiar()
{
    char c;
    char archi[50];
    int i=0;
    int fd=0;
    int fd2;
    char num='0';
    char destino[50]="/home/emm/11_Copias/*.txt"; //Se da una direccion destino previa
    int con=1;

    while(con==1)
    {
        system("clear");
        printf("\nIngrese el archivo que desea copiar: ");
        scanf("%s", archi);

        fd = open(archi,O_RDONLY);

        if(fd!=-1) //Si el archivo existe...
```

```

    {
        destino[20]=num;

        fd2 = open(destino,O_WRONLY|O_CREAT,S_IRUSR|S_IWUSR); //Se crea un nuevo archivo
        while(read(fd,&c,sizeof(c))!=0)
        {
            write(fd2,&c,sizeof(c)); //se copia y ...
        }

        printf("\nArchivo fuente en: %s \nArchivo copiado en: %s\n", archi,destino);
        num++;
        close(fd);
        close(fd2);
    } //... esto se repite para cada archivo que se desee copiar
    else
        printf("\n El archivo escrito no existe en este directorio \n");

    printf("\nDesea introducir otro archivo? Si = 1 , No = 0 : ");
    scanf("%d",&con);
}
return 0;
}

int menu(int opc)
{
    printf("\n\nQue desea hacer?\n");
    printf("1. Mostrar contenido de un archivo\n");
    printf("2. Copiar archivo al directorio previamente establecido\n");
    printf("3. Salir\n");
    return 0;
}

//Funci n principal
int main()
{
    //Declaramos variables
    int opc;

    //O_RDONLY abre el archivo salida.txt en modo lectura

    do{
        menu(opc);
        scanf("%d", &opc);
        system("clear");

        switch(opc)
        {
            case 1:
                Mostrar();
                break;

            case 2:
                Copiar();
                break;
        }
    }

```

```

        default:
            printf("\nADIOS\n");
            break;

    }
} while(opc<3);
}

```

2.3.2. Windows

Creación de Archivos

```

#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <windows.h>
#include <fcntl.h>
#include <time.h>
#include <stdlib.h>

int main(int argc, char const *argv[])
{
    char name[]="Archivo .txt";
    char content[]="Hola Mundo";
    char dir[100];
    char aux[100];
    int NFile,IdFile,i;
    HANDLE file;// Apuntador al archivo
    printf("Give me the name of the Directory\n");
    scanf("%s",dir);
    if(!(CreateDirectory(dir,NULL))) {
        printf("Directory don't create\n");
    }
    else {
        NFile=rand()%11;
        for (i = 0; i < NFile; ++i)
        {
            strcpy(aux, ".");
            strcpy(aux, dir);
            name[7]=i+'0';
            strcat(aux, "/");
            strcat(aux, name);
            printf("%s\n", aux);
            file=CreateFile(aux, GENERIC_WRITE, 0, NULL, CREATE_NEW, FILE_ATTRIBUTE_NORMAL, NULL);
            WriteFile(file, content, sizeof(content), NULL, NULL);
            strcpy(aux, "\0");
        }
    }
    return 0;
}

```

Enlistado de Archivos

```

#include <stdio.h>
#include <stdlib.h>
#include <windows.h>
#include <sys/types.h>
#include <stdint.h>
#include <time.h>
#include <dirent.h>
#include <sys/stat.h>
#include <locale.h>
/* run this program using the console pauser or add your own getch, system("pause")
void comenzar();
int main(int argc, char *argv[]) {
    printf("Ingresa el directorio");
    comenzar();
    return 0;
}
void comenzar(){
    char directorio[20];
    struct dirent *dp;
    struct stat statbuf;
    struct tm *tm;
    struct group *grp;
    struct passwd *pwd;
    off_t tamanio;
    char datestring[30];

    scanf("%s",directorio);
    DIR *dirp=opendir(directorio);
    tm = localtime(&statbuf.st_mtime);
    int i=-1;
    while((dp=readdir(dirp))!=NULL){
        if(stat(dp->d_name,&statbuf)==-1)
        {
            printf("%s\t", dp->d_name);
            printf("TAMANIO: ");
            printf(" %lld\t",statbuf.st_size);
            printf("Hora de acceso: ");
            tm = localtime(&statbuf.st_mtime);
            strftime(datestring, sizeof(datestring), "%x - %I:%M%p", tm);
            printf("%s",datestring);
            printf("\n");
        }
    }
    system("pause");
}

```

Mostrar contenido de un archivo y copiado de archivos

```
#include <windows.h>
#include <stdio.h>
#include <string.h>

int Copiar()
{
    int con=1;
    char num='0';
    char destino[50]="F:\\S0\\Copias_11\\*.txt"; //Directorio destino previamente creado
    char archi[50];
        while(con==1){
            system("cls");
            printf("\nEscriba el directorio del archivo que quiere copiar seguido de su nombre:");
            scanf("%s", &archi);

            destino[16]=num; //El nombre del archivo copiado sera un numero

int cp= CopyFile(archi,destino,0); //Copiamos lo encontrado en el archivo origen y
    if(!cp)
    {
        printf("\nError al copiar el archivo\n");
    }
    else
    {
        printf("\nArchivo fuente en: %s \nArchivo copiado en: %s\n", archi,destino);
        num++;
    }

    printf("\nDesea copiar otro archivo?  SI = 1 , NO = 0 : ");
    scanf("%d", &con);
}
}

int menu(opc)
{
    printf("\n\nQue desea hacer?\n");
    printf("1. Mostrar contenido de un archivo\n");
    printf("2. Copiar archivo al directorio previamente establecido\n");
    printf("3. Salir\n");
    return 0;
}

int main(){

    //Declaramos variables
    int opc;
    int con=1;
    HANDLE hFile;
    char c;
    int i;
    char archi[60];
    DWORD taman=1000;
    DWORD bytesRead;
    char buff[4096];
```



```
do{
menu(opc);
scanf("%d", &opc);
system("cls");

switch(opc)
{
    case 1:

        printf("\nEscriba cual es el archivo que desea abrir para ver su contenido, recuerde el nombre\n");
        scanf("%s", archi);
        hFile = CreateFile(archi, GENERIC_READ|GENERIC_WRITE, FILE_SHARE_READ, NULL, OPEN_EXISTING, 0, NULL);
        printf("\n");

        if(hFile != INVALID_HANDLE_VALUE)
        {
            ReadFile(hFile, buff, 4096, &bytesRead, NULL); //Leemos el archivo y almacenamos lo leído en buff

            for(i=0; i<=strlen(buff); i++) //Mostramos en pantalla el mensaje leído de buff
                printf("%c", buff[i]);

            //Se cierra el archivo y la opcion del menu .
            close(hFile);

            for(i=0; i<strlen(buff); i++)
                buff[i]=' ';
        }
        else
            printf("\nProblema al intentar visualizar archivo\n");

        break;

    case 2:

        Copiar();
        break;

    default:
        printf("\nADIÓS\n");
        break;
}
} while(opc<3);
return 0;
}
```

3. Observaciones.

La principal observación que encontramos en el desarrollo de esta práctica es la diferencia del tipo de sistema de archivos en los dos sistemas operativos, Linux y Windows, lo cual ocasiona que existan algunas diferencias entre la interacción con las interfaces de llamadas al sistema entre los sistemas operativos, como fue el que el sistema operativo windows no permite un cambio en los permisos de usuario.

Por otro lado, como sabemos las llamadas al sistema son la interfaz entre la capa de servicios y la capa de aplicación, por lo que al ser diferentes las llamadas al sistema entre sistemas operativos en consecuencia son diferentes los servicios o la forma en que se interactúa en la capa de aplicación.

4. Análisis Crítico.

Ambos sistemas operativos manejan llamadas a su sistema que nos permiten un acercamiento más profundo, íntimo vaya, al funcionamiento de cada sistema mediante comandos de cada uno, con los cuales pudimos manejar archivos (crearlos, escribir en ellos, obtener información, etc.); sin embargo, es desde linux donde se tiene más libertad de utilizar este tipo de comandos, donde hay mayor diversidad de ellos, puesto que por naturaleza está diseñado para usuarios expertos en la utilización de un equipo de cómputo, ya que en windows algunas funcionalidades están en desuso o cuesta más trabajo implementarlas; en pocas palabras existe mayor restricción sobre el manejo de los archivos desde windows.

5. Conclusiones.

Se han cumplido los objetivos de esta práctica debido a que se implementaron con éxito las distintas llamadas al sistemas, tanto en linux como en windows, familiarizándonos con el uso de la interfaz de comandos respectiva. Hemos creado otra perspectiva sobre la funcionalidad de los sistemas operativos, manejando archivos como quizá pocos usuarios lo hayan experimentado, ya que las llamadas al sistemas nos permiten un control más íntimo de los archivos (de la información almacenada en un equipo de cómputo), siendo linux donde existe mayor control que en windows.