

Practica 2

Instituto Politecnico Nacional Escuela Superior de Computo

Sistemas operativos

Interfaz de llamadas al sistema

Ethan Jezreel Lopez Torres
Gonzaga Martínez José Alberto
Sebastian Absalon Cortes

Marco Teorico

Sistemas Operativos Un sistema operativo es el programa que, después de ser cargado inicialmente en la computadora por un programa de arranque, administra todos los demás programas de aplicación en una computadora. Los programas de aplicación hacen uso del sistema operativo al realizar solicitudes de servicios a través de una interfaz de programa de aplicación definida. Además, los usuarios pueden interactuar directamente con el sistema operativo a través de una interfaz de usuario, como una interfaz de línea de comandos o una interfaz de usuario gráfica.

Windows Se conoce como Windows, MS Windows o Microsoft Windows a una familia de sistemas operativos para computadores personales, teléfonos inteligentes y otros sistemas informáticos, creados y comercializados por la empresa norteamericana Microsoft para diversos soportes de arquitectura de sistemas. La primera aparición de Windows ocurrió en 1985, como un paso adelante en la modernización del MS-DOS hacia los entornos gráficos de usuario (GUI), y desde entonces se ha convertido en el sistema operativo más utilizado del mundo, copando prácticamente la totalidad de la

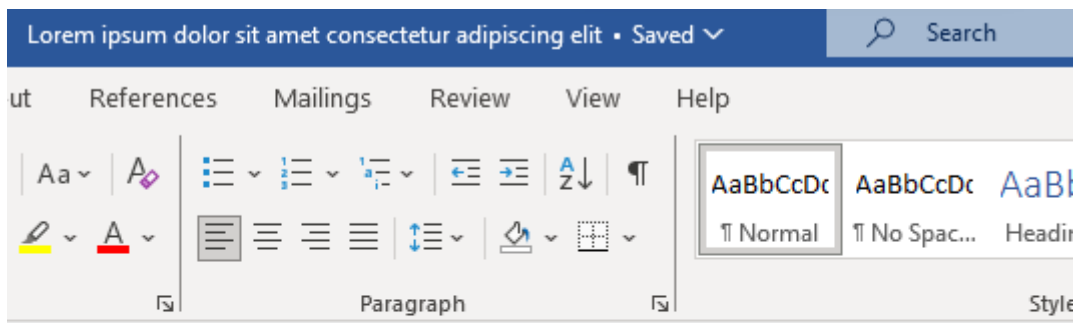
cuota de mercado disponible (90%) durante años. Windows ofreció a sus usuarios una creciente variedad de versiones disponibles y actualizadas del programa, con diferencias notorias en cuanto a su aspecto, estabilidad y potencias. La incorporación de Internet permitió, además, la actualización automática del software en cualquier parte del mundo.

Linux Linux es un sistema operativo open source. En 1991, Linus Torvalds lo diseñó y creó a modo de pasatiempo. Mientras estaba en la universidad, intentó crear una versión open source, alternativa y gratuita del sistema operativo MINIX, que a su vez se basaba en los principios y el diseño de Unix. Ese pasatiempo logró convertirse en el SO con la mayor base de usuarios, el más usado en los servidores de Internet disponibles públicamente y en el único utilizado en las 500 supercomputadoras más rápidas.

Archivos en windows y linux

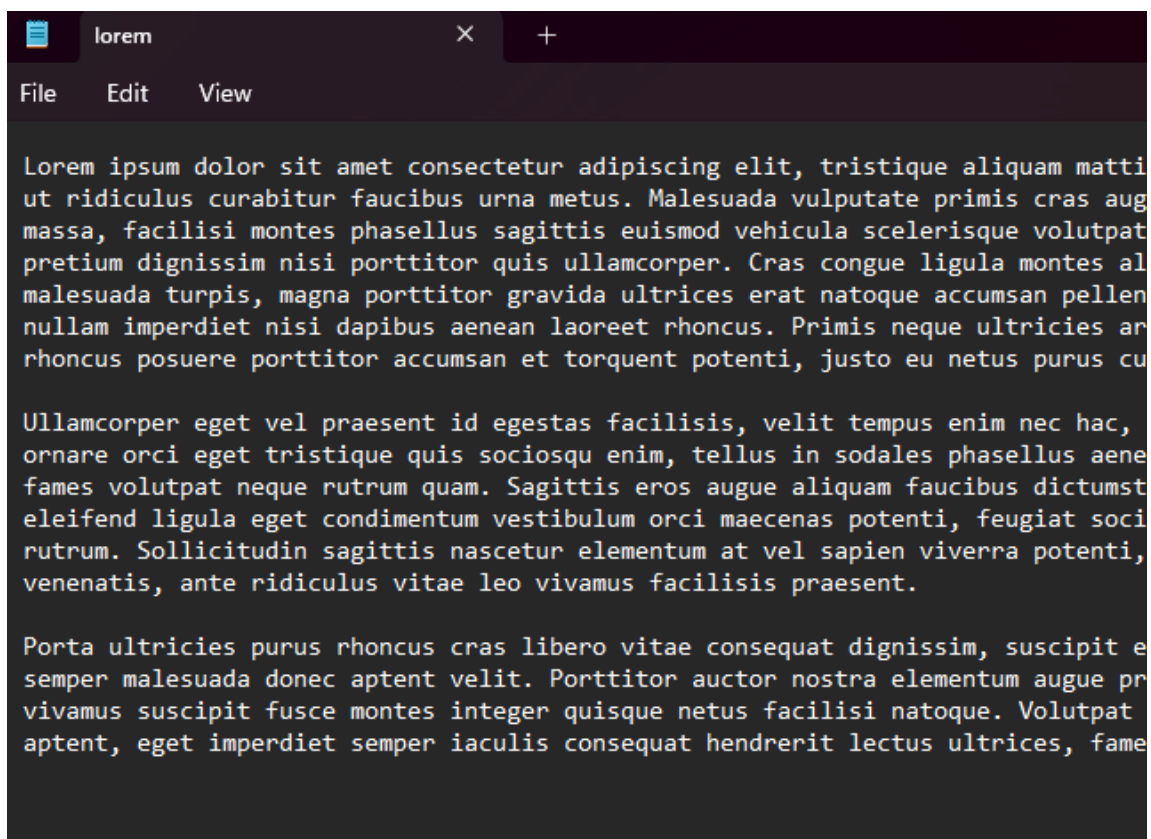
Gracias al tiempo que estos sistemas operativos han estado activos en el mundo y a la cantidad de usuarios que les dan soporte día con día y crean mas y mejores herramientas para poder ocuparlos, windows y linux son cada día mas compatibles, esto nos permite en muchas ocasiones poder trabajar algo en un sistema operativo y cambairnos a otro sin mayor problema, sin embargo esto no siempre sucede, en alguans ocasiones y mas si no tenemos experiencia con el tema, podemos tener ciertos errores al momento de trabajar con ambos sistemas operativos, a continuacion veremos algunos ejemplos de ello.

Primeramente en windows creamos dos archivos, uno tipo dox y un tipo txt.



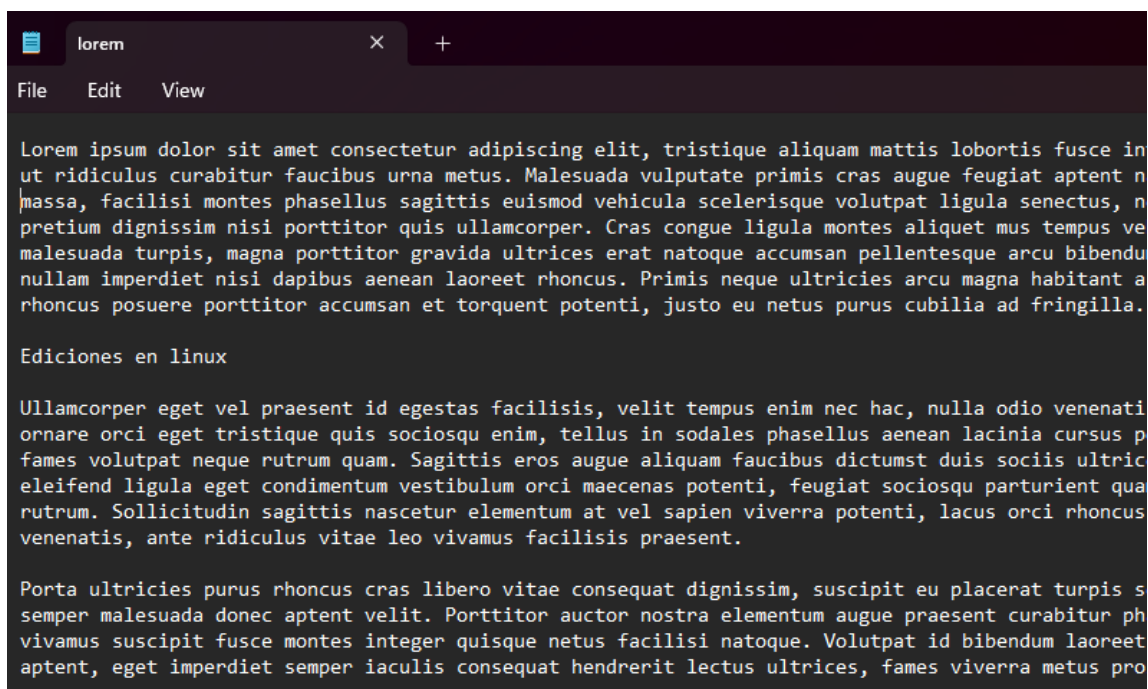
parturient quam suscipit volutpat t
Sollicitudin sagittis nascetur eleme
sapien viverra potenti, lacus orci rho
accumsan cursus venenatis, ante ridic
vivamus facilisis praesent.

Porta ultricies purus rhoncus cras
consequat dignissim, suscipit eu place
mollis sem, diam nunc semper male
aptent velit. Porttitor auctor nostr
augue praesent curabitur pharetra d
vivamus suscipit fusce montes integer

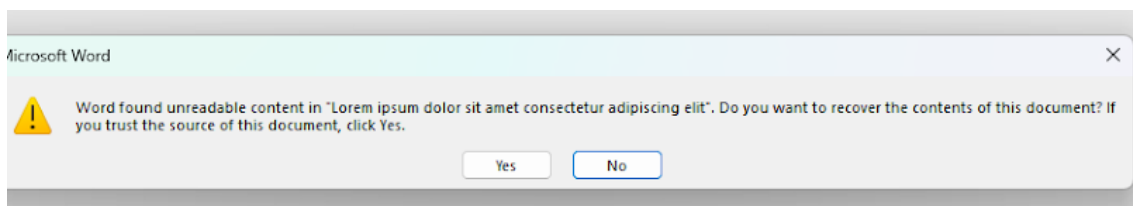


A continuacion, cambiando de sistema operativo a Ubuntu, podemos editar estos documentos, sin embargo, cometemos un error al editar el archivo tipo dox en una aplicacion que no soporta estas ediciones.

De vuelta en windows, podemos darnos cuenta de que por un lado, estas ediciones no causaron conflicto alguno en el caso del archivo en txt.



Sin embargo, no podemos decir lo mismo de nuestro archivo tipo dox.



De todo esto podemos destacar que es importante verificar el tipo de ediciones que hacemos en cada sistema operativo, así como los programas con los que esto se realiza, ya que muchos de los editores de texto no son compatibles unos con otros, esto provoca que los archivos se editen de forma diferente y no puedan ser leídos por todos los editores.

Llamadas al sistema de Linux

El Linux nos podemos encontrar con llamadas al sistema que nos permiten realizar operaciones directas con el sistema operativo. A continuación enlistamos algunas de ellas y sus funciones correspondientes.

Comando	Funcionamiento
open	Abre un archivo
close	Cierra el descriptor de un archivo
read	Lee mediante el descriptor de un archivo
write	Escribe mediante el descriptor de un archivo

creat	Abre un archivo y en caso de que este no exista lo crea
lseek	Reposiciona el apuntador de lectura de un programa
access	Permite revisar si el usuario actual tiene permisos para un archivo
stat	Muestra el nomrbe del archivo o el status de un archivo
chmod	Permite cambiar los permisos de lectura, escritura y ejecucion
chown	Permite cambiar la propiedad de un archivo a otro usuario
fcntl	Permite manipular el descriptor de un archivo
chdir	Cambia el directorio de trabajo actual en la consola
mkdir	Permite crear un nuevo directorio de archivos
opendir	Permite abrir un directorio de archivos
readdir	Permite leer el contenido de un directorio de arvhivos

Llamadas al sistema de Windows

Comando	Funcion
OpenFile	Crea, abre, vuelve a abrir o elimina un archivo.
CloseFile	Para utilizar los recursos del sistema operativo de manera eficiente, una aplicación debe cerrar los archivos cuando ya no sean necesarios mediante la función CloseHandle. Si un archivo está abierto cuando finaliza una aplicación, el sistema lo cierra automáticamente.
CloseHandle	***
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.
WriteFile	Escribe datos en el archivo o dispositivo de entrada/salida (E/S) especificado.
CreateFileA	Crea o abre un archivo o dispositivo de E/S. Los dispositivos de E/S más utilizados son los siguientes: archivo, flujo de archivos, directorio, disco físico, volumen, búfer de consola, unidad de cinta, recurso de comunicaciones, ranura de correo y canalización.
SetFilePointer	Mueve el puntero del archivo del archivo especificado. Esta función almacena el puntero del archivo en dos valores LARGO.
CreateDirectoryA	Crea un nuevo directorio. Si el sistema de archivos subyacente admite la seguridad de archivos y directorios, la función aplica un descriptor de seguridad específico al nuevo directorio.

SetCurrentDirectory	Cambia el directorio actual para el proceso actual.
GetFileAttributesA	Recupera atributos del sistema de archivos para un archivo o directorio específico.
SetFileAttributesA	Establece los atributos de un archivo o directorio.
FindFirstFileA	Busca en un directorio un archivo o subdirectorio con un nombre que coincida con un nombre específico (o un nombre parcial si se utilizan comodines).
FindNextFileA	Continúa una búsqueda de archivos desde una llamada anterior a FindFirstFile

Programas Linux

Serie aleatoria de archivos.

```

1 #define _POSIX_SOURCE
2 #include <dirent.h>
3 #include <errno.h>
4 #include <stdio.h>
5 #include <string.h>
6 #include <time.h>
7 #include <stdlib.h>
8 #include <unistd.h>
9 #include <fcntl.h>
10 #include <sys/types.h>
11 #include <sys/stat.h>
12 #undef _POSIX_SOURCE
13
14 int buscar(const char *arch)
15 {
16     DIR *dirp;
17     struct dirent *buscador;
18
19     if ((dirp = opendir(".")) == NULL) {
20         perror("couldn't open '.');
21         return 0;
22     }
23
24     do {
25         errno = 0;
26         if ((buscador = readdir(dirp)) != NULL) {
27             if (strcmp(buscador->d_name, arch) != 0)
28                 continue;
29             (void) printf("found %s\n", arch);
30             (void) closedir(dirp);
31             return 1;
32         }
33     } while (buscador != NULL);
34
35     if (errno != 0)
36         perror("error reading directory");
37     else
38         (void) printf("Se creo el archivo %s\n", arch);
39     (void) closedir(dirp);
40     return 0;
41 }
42
43
44
45 int main( )
46 {
47     int i, longitud=5, num_archivos=5;
48     char ruta[100];
49     struct stat st = {0};
50     printf("Teclea el nuevo directorio\n");
51     while(scanf("%s", ruta)){
52         if (stat(ruta, &st) == -1) {
53             mkdir(ruta, 0700);
54             break;
55         }else{
56             printf("El directorio ya existe intento de nuevo\n");
57         }
58     }
59
60     int path=chdir(ruta);
61     char contenido[5][10] = {"archivo1", "archivo2", "archivo3", "archivo4", "archivo5"};
62     srand(time(NULL));
63     for(int z=0 ; z<num_archivos; z++){
64         char fn[]="holas.txt";
65         for( int i = 0; i < longitud; i++){
66             fn[i] = 'a' + rand()%25; // empieza en a y termina en z
67         }
68
69         if(buscar(fn)==0){
70             int fd = creat(fn, S_IRUSR | S_IWUSR);
71             fd = open(fn, O_WRONLY);
72             write(fd, contenido[z], strlen(contenido[z]));
73             close(fd);
74         }else{
75             i--;
76         }
77     }
78
79 }
80
81 return (0);
82 }

```








```

suzu@cozzy:~/Documents/operatingSystems/practicas/practica2/Programas_Linux$ g
cc aleatorio.c -o aleatorio
suzu@cozzy:~/Documents/operatingSystems/practicas/practica2/Programas_Linux$ .
/aleatorio
Teclea el nuevo directorio
45
Se creo el archivo qmnnw.txt
Se creo el archivo wagjg.txt
Se creo el archivo dqybc.txt
Se creo el archivo wsxpg.txt
Se creo el archivo hflop.txt
suzu@cozzy:~/Documents/operatingSystems/practicas/practica2/Programas_Linux$

```

Los archivos creados.

Nombre	Tipo	Tamaño comprimido	Protegido ...	Tamaño
 dqybc	Documento de texto	1 KB	No	1 KB
 hflop	Documento de texto	1 KB	No	1 KB
 qmnnw	Documento de texto	1 KB	No	1 KB
 wagjg	Documento de texto	1 KB	No	1 KB
 wsxpg	Documento de texto	1 KB	No	1 KB

Lista de archivos creados, mostrando su tamaño, fecha y hora de acceso.

```
1  #include <dirent.h>
2  #include <errno.h>
3  #include <stdio.h>
4  #include <sys/types.h>
5  #include <sys/stat.h>
6  #include <time.h>
7  #include <unistd.h>
8  #include <fcntl.h>
9  #include <stdlib.h>
10 #include <sys/sysmacros.h>
11
12
13 int main()
14 {
15     struct stat sb;
16     DIR *d;
17     struct dirent *dir;
18     char ruta[100];
19     printf("Teclea la ruta\n");
20     scanf("%s", ruta);
21     int path=chdir(ruta);
22     d = opendir(".");
23
24     if (d) {
25         while ((dir = readdir(d)) != NULL) {
26             printf("-> %s\n", dir->d_name);
27
28             if (lstat(dir->d_name, &sb) == -1) {
29                 perror("lstat");
30                 exit(EXIT_FAILURE);
31             }
32
33             printf("Tamano:                %lld bytes\n", (long long) sb.st_size);
34             printf("Ultimo cambio de estado: %s", ctime(&sb.st_ctime));
35             printf("Ultimo acceso al archivo: %s", ctime(&sb.st_atime));
36             printf("Ultima modificacion:      %s", ctime(&sb.st_mtime));
37         }
38         closedir(d);
39     }
40
41
42     exit(EXIT_SUCCESS);
43 }
```

```
suzu@cozzy:~/Documents/operatingSystems/practicass/practica2/Programas_Linux$ gcc listar_archivos.c -o listar
suzu@cozzy:~/Documents/operatingSystems/practicass/practica2/Programas_Linux$ ./listar
Teclea la ruta
./45
-> wsxpg.txt
Tamano: 8 bytes
Ultimo cambio de estado: Mon Oct 2 18:58:04 2023
Ultimo acceso al archivo: Mon Oct 2 18:58:20 2023
Ultima modificacion: Mon Oct 2 18:58:04 2023
-> hflop.txt
Tamano: 8 bytes
Ultimo cambio de estado: Mon Oct 2 18:58:04 2023
Ultimo acceso al archivo: Mon Oct 2 18:58:20 2023
Ultima modificacion: Mon Oct 2 18:58:04 2023
-> wagjg.txt
Tamano: 8 bytes
Ultimo cambio de estado: Mon Oct 2 18:58:04 2023
Ultimo acceso al archivo: Mon Oct 2 18:58:20 2023
Ultima modificacion: Mon Oct 2 18:58:04 2023
-> copias
Tamano: 6 bytes
Ultimo cambio de estado: Mon Oct 2 19:01:51 2023
Ultimo acceso al archivo: Mon Oct 2 19:02:06 2023
Ultima modificacion: Mon Oct 2 19:01:51 2023
-> outputAleatorio.zip
Tamano: 942 bytes
Ultimo cambio de estado: Mon Oct 2 18:59:19 2023
Ultimo acceso al archivo: Mon Oct 2 18:59:34 2023
Ultima modificacion: Mon Oct 2 18:59:19 2023
-> dqybc.txt
Tamano: 8 bytes
Ultimo cambio de estado: Mon Oct 2 18:58:04 2023
Ultimo acceso al archivo: Mon Oct 2 18:58:20 2023
Ultima modificacion: Mon Oct 2 18:58:04 2023
-> .
Tamano: 4096 bytes
Ultimo cambio de estado: Mon Oct 2 19:01:51 2023
Ultimo acceso al archivo: Mon Oct 2 19:01:51 2023
Ultima modificacion: Mon Oct 2 19:01:51 2023
```

Programa que copie uno o más de los archivos creados a un directorio previamente establecido.

```

1 #include <stdio.h>
2 #include <windows.h>
3 #include <string.h>
4 #include <time.h>
5 #include <sys/stat.h>
6 #include <stdlib.h>
7 #include <dirent.h>
8 #include <sys/types.h>
9 int main(){
10     char nombreRuta[20][50], nombreRutaGuardada[60], contenido[20][200], nombreArchivos[20][20];
11     HANDLE archivo;
12     DWORD tamaño=20;
13     DIR *dir;
14     struct dirent *sd;
15     struct stat buf;
16     int i=0, pruebaArchivo, numeroArchivo;
17     char ruta[100], rutaGuardada[100];
18     printf("\nIngresa la ruta de la carpeta que quieres escanear\n");
19     scanf("%s", ruta);
20     dir= opendir(ruta);
21     if(dir==NULL){
22         exit(1);
23     }
24     while((sd=readdir(dir)) != NULL){
25         strcpy(nombreArchivos[i], sd->d_name);
26         strcpy(nombreRuta[i], ruta);
27         strcat(nombreRuta[i], "\\");
28         strcat(nombreRuta[i], sd->d_name);
29         archivo=CreateFile(nombreRuta[i], GENERIC_READ, 0, NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);
30         if(pruebaArchivo=ReadFile(archivo, contenido[i], tamaño, 0, NULL)!= 0){
31             printf("id: %d\tNombre: %s\n", i+1, nombreArchivos[i]);
32             i++;
33         }
34     }
35     char pregunta[3];
36     printf("Quieres ver el contenido de algun archivo\n");
37     scanf("%s", pregunta);
38     if(strcmp(pregunta, "no")==0 || strcmp(pregunta, "NO")==0 || strcmp(pregunta, "No")==0;
39     else{
40         printf("teclea el id de los archivos que quieres ver su contenido (para terminar teclea 0)\n");
41         while(scanf("%d", &numeroArchivo) && numeroArchivo && numeroArchivo<(i+1)){
42             printf("Contenido: %s\n", contenido[numeroArchivo-1]);
43         }
44     }
45     printf("Quieres copiar algun archivo en otra carpeta?\n");
46     scanf("%s", pregunta);
47     if(strcmp(pregunta, "no")==0 || strcmp(pregunta, "NO")==0 || strcmp(pregunta, "No")==0
48     exit(1);
49     else{
50         printf("Introduce la ruta a donde quieres copiarlos\n");
51         scanf("%s", rutaGuardada);
52         printf("teclea los numeros de archivos que quieres copiar (Para dejar de copiar teclea 0)\n");
53         scanf("%d", &numeroArchivo);
54         while(numeroArchivo){
55             if(numeroArchivo>i){
56                 printf("ese numero no pertenece a ningun archivo\n");
57             }else{
58                 strcpy(nombreRutaGuardada, rutaGuardada);
59                 strcat(nombreRutaGuardada, "\\");
60                 strcat(nombreRutaGuardada, nombreArchivos[numeroArchivo-1]);
61                 archivo=CreateFile(nombreRutaGuardada, GENERIC_WRITE, FILE_SHARE_READ, NULL, CREATE_ALWAYS, FILE_ATTRIBUTE_NORMAL, NULL);
62                 if(WriteFile(archivo, contenido[numeroArchivo-1], strlen(contenido[numeroArchivo-1]), 0, NULL)!=0)printf("Archivo Copiado Exitosamente\n");
63             }
64             fflush(stdin);
65             scanf("%d", &numeroArchivo);
66         }
67     }
68     return 0;
69
70 }

```

```

suzu@cozzy:~/Documents/operatingSystems/practicas/practica2/Programas_Linux$ ./copiar
Teclea 1 para visualizar un archivo
        2 para copiar un archivo
        3 para salir
1
Teclea la direccion del archivo a visualizar
./45/hflop.txt
-----
archivo5
-----
Teclea 1 para visualizar un archivo
        2 para copiar un archivo
        3 para salir
2
Teclea la direccion del archivo a copiar
./45/hflop.txt

Teclea la direccion del archivo de destino
./45/copias
Teclea 1 para visualizar un archivo
        2 para copiar un archivo
        3 para salir
3
El programa ha terminado exitosamente
suzu@cozzy:~/Documents/operatingSystems/practicas/practica2/Programas_Linux$

```

Programas en Windows

Un programa en C que crea una serie aleatoria de archivos.

```

1 #include <stdio.h>
2 #include <windows.h>
3 #include <string.h>
4 #include <time.h>
5 int main(){
6     int w;
7     char gg[200],nombreArchivo[10];
8     HANDLE hfile;
9     char cadenas[10][15]= {"silla","pastel","forma","casual","intenso","combinacio","memoria","contexto","portable","contador"};
10    char ruta[100];
11    printf("\nIngresa la ruta de la carpeta donde quieres crear los archivos\n\n");
12    scanf("%s",ruta);
13    strcat(ruta,"/NuevoDirectorio/");
14    CreateDirectory(ruta,NULL);
15    srand(time(NULL));
16    int numArchivos=rand()%10;
17    if(numArchivos==0)numArchivos=1;
18    printf("\nSe han creado %d archivos de nombres:\n\n",numArchivos);
19    for(int i=0;i<numArchivos;i++){
20        strcpy(gg,ruta);
21        for(int j=0;j<3;j++){
22            nombreArchivo[j]= 'a'+rand()%25;
23        }
24        printf("\t%s.txt\n",nombreArchivo);
25        strcat(gg,nombreArchivo);
26        strcat(gg,".txt");
27        hfile=CreateFile(gg,GENERIC_WRITE,FILE_SHARE_READ,NULL,CREATE_ALWAYS,FILE_ATTRIBUTE_NORMAL,NULL);
28        w=WriteFile(hfile,cadenas[i],strlen(cadenas[i]),0,NULL);
29    }
30    CloseHandle(hfile);
31    return 0;
32 }
33 }
34

```

```
C:\Users\alber\Desktop\SISTEMAS OPERATIVOS>gcc programa_p8.c
C:\Users\alber\Desktop\SISTEMAS OPERATIVOS>a
Ingresa la ruta de la carpeta donde quieres crear los archivos
C:\Users\alber\Desktop\Carpeta
Se han creado 5 archivos de nombres:
    jyj.txt
    olb.txt
    een.txt
    lbl.txt
    qca.txt
```

En el siguiente programa ingrese la ruta C:\Users, mostrando en la terminal su tamaño, hora de acceso y nombre.

```

1  #include <stdio.h>
2  #include <windows.h>
3  #include <string.h>
4  #include <time.h>
5  #include <sys/stat.h>
6  #include <stdlib.h>
7  #include <dirent.h>
8  #include <sys/types.h>
9  int main(){
10  DIR *dir;
11  struct stat buf;
12  struct dirent *sd;
13  int i=0;
14  char ruta[100],atime[100],rutaArchivo[150];
15  long size;
16  char archivito[20][30];
17  printf("\nIngresa la ruta de la carpeta que quieres escanear\n");
18  scanf("%s",ruta);
19  system("CLS");
20  dir= opendir(ruta);
21  if(dir==NULL){
22      exit(1);
23  }
24  while((sd=readdir(dir)) != NULL){
25      if(i>=2)
26      {strcpy(rutaArchivo,ruta);
27       strcat(rutaArchivo,"/");
28       strcat(rutaArchivo,sd->d_name);
29       stat(rutaArchivo,&buf);
30       strcpy(atime,ctime(&buf.st_atime));
31       printf("Nombre: %s\t Tamano en bytes: %lld\t Ultima hora de acceso: %s",sd->d_name,(long long)buf.st_size,atime);}
32      i++;
33  }
34  closedir(dir);
35  printf("\n");
36  return 0;
37  }

```

```

Nombre: alber      Tamano en bytes: 0      Ultima hora de acceso: Mon Oct 02 14:43:51 2023
Nombre: All Users  Tamano en bytes: 0      Ultima hora de acceso: Fri May 06 23:41:31 2022
Nombre: Default    Tamano en bytes: 0      Ultima hora de acceso: Mon Oct 02 15:17:16 2023
Nombre: Default User Tamano en bytes: 0      Ultima hora de acceso: Fri May 06 23:41:31 2022
Nombre: defaultuser1000000 Tamano en bytes: 0      Ultima hora de acceso: Mon Oct 02 15:08:13 2023
Nombre: desktop.ini Tamano en bytes: 174  Ultima hora de acceso: Mon Oct 02 14:43:53 2023
Nombre: Public     Tamano en bytes: 0      Ultima hora de acceso: Mon Oct 02 14:43:53 2023

```

Como ultimo muestro el funcionamiento para el programa del punto numero 11 de la práctica, en donde se muestra el contenido de los archivos seleccionados y los copie en un nuevo directorio.

```

1 #include <stdio.h>
2 #include <windows.h>
3 #include <string.h>
4 #include <time.h>
5 #include <sys/stat.h>
6 #include <stdlib.h>
7 #include <dirent.h>
8 #include <sys/types.h>
9 int main(){
10     char nombreRuta[20][50], nombreRutaGuardada[60], contenido[20][200], nombreArchivos[20][20];
11     HANDLE archivo;
12     DWORD tamaño=20;
13     DIR *dir;
14     struct dirent *sd;
15     struct stat buf;
16     int i=0, pruebaArchivo, numeroArchivo;
17     char ruta[100], rutaGuardada[100];
18     printf("\nIngresa la ruta de la carpeta que quieres escanear\n");
19     scanf("%s", ruta);
20     dir= opendir(ruta);
21     if(dir==NULL){
22         exit(1);
23     }
24     while((sd=readdir(dir)) != NULL){
25         strcpy(nombreArchivos[i], sd->d_name);
26         strcpy(nombreRuta[i], ruta);
27         strcat(nombreRuta[i], "\\");
28         strcat(nombreRuta[i], sd->d_name);
29         archivo=CreateFile(nombreRuta[i], GENERIC_READ, 0, NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);
30         if(pruebaArchivo=ReadFile(archivo, contenido[i], tamaño, 0, NULL)!= 0){
31             printf("id: %d\tNombre: %s\n", i+1, nombreArchivos[i]);
32             i++;
33         }
34     }
35     char pregunta[3];
36     printf("Quieres ver el contenido de algun archivo\n");
37     scanf("%s", pregunta);
38     if(strcmp(pregunta, "no")==0 || strcmp(pregunta, "NO")==0 || strcmp(pregunta, "No")==0;
39     else{
40         printf("teclea el id de los archivos que quieres ver su contenido (para terminar teclea 0)\n");
41         while(scanf("%d", &numeroArchivo) && numeroArchivo && numeroArchivo<(i+1)){
42             printf("Contenido: %s\n", contenido[numeroArchivo-1]);
43         }
44     }
45     printf("Quieres copiar algun archivo en otra carpeta?\n");
46     scanf("%s", pregunta);
47     if(strcmp(pregunta, "no")==0 || strcmp(pregunta, "NO")==0 || strcmp(pregunta, "No")==0
48     exit(1);
49     else{
50         printf("Introduce la ruta a donde quieres copiarlos\n");
51         scanf("%s", rutaGuardada);
52         printf("teclea los numeros de archivos que quieres copiar (Para dejar de copiar teclea 0)\n");
53         scanf("%d", &numeroArchivo);
54         while(numeroArchivo){
55             if(numeroArchivo>i){
56                 printf("ese numero no pertenece a ningun archivo\n");
57             }else{
58                 strcpy(nombreRutaGuardada, rutaGuardada);
59                 strcat(nombreRutaGuardada, "\\");
60                 strcat(nombreRutaGuardada, nombreArchivos[numeroArchivo-1]);
61                 archivo=CreateFile(nombreRutaGuardada, GENERIC_WRITE, FILE_SHARE_READ, NULL, CREATE_ALWAYS, FILE_ATTRIBUTE_NORMAL, NULL);
62                 if(WriteFile(archivo, contenido[numeroArchivo-1], strlen(contenido[numeroArchivo-1]), 0, NULL)!=0)printf("Archivo Copiado Exitosamente\n");
63             }
64             fflush(stdin);
65             scanf("%d", &numeroArchivo);
66         }
67     }
68     return 0;
69 }
70 }

```



```
C:\Users\alber\Desktop\SISTEMAS OPERATIVOS>gcc programa_p11.c
```

```
C:\Users\alber\Desktop\SISTEMAS OPERATIVOS>a
```

Ingresa la ruta de la carpeta que quieres escanear

```
C:\Users\alber|
```

```
id: 13          Nombre: ntuser.ini
Quieres ver el contenido de algun archivo
si
teclea el id de los archvos que quieres ver su contenido (para terminar teclea 0)
1
Contenido:      {
  "$schema": " ./noD~a

1
Contenido:      {
  "$schema": " ./noD~a
2
Contenido:      ls
nkdir test
nkdir

0
Quieres copiar algun archivo en otra carpeta?
si
Introduce la ruta a donde quieres copiarlos
no
teclea los numeros de archivos que quieres copiar (Para dejar de copiar teclea 0)
0

C:\Users\alber\Desktop\SISTEMAS OPERATIVOS>a.exe|
```

Aqui el proceso que se realizo para copiar un archivo a un nuevo directorio.

```

C:\Users\alber\Desktop\SISTEMAS OPERATIVOS>a

Ingresa la ruta de la carpeta que quieres escanear
C:\Users\alber\Desktop\EJEMPLO
id: 1          Nombre: TEXTO_PRACTICA.txt
Quieres ver el contenido de algun archivo
si
teclea el id de los archivos que quieres ver su contenido (para terminar teclea 0)
1
Contenido:     Este es un ejemplo pD~a
0
Quieres copiar algun archivo en otra carpeta?
si
Introduce la ruta a donde quieres copiarlos
C:\Users\alber\Desktop\
teclea los numeros de archivos que quieres copiar (Para dejar de copiar teclea 0)
2
ese numero no pertenece a ningun archivo
1
Archivo Copiado Exitosamente
0

```

Muestro que el archivo fue copiado sin problemas en el nuevo directorio.

```

20/09/2023 03:00 p. m. <DIR> ORDENAR CARPETAS Y ARCHIVOS
02/10/2023 03:44 p. m. <DIR> SISTEMAS OPERATIVOS
02/10/2023 08:43 a. m. <DIR> TECNOLOGIAS WEB
02/10/2023 03:45 p. m. 23 TEXTO_PRACTICA.txt
14/08/2023 06:00 p. m. 222 Wallpaper Engine.unl

```

Conclusiones

Es importante conocer las herramientas necesarias y las funciones disponibles por parte del sistema para la manipulación de los medios, permisos y archivos, sin este tipo de llamadas al sistema sería más complicado realizar tareas simples como el manejo de archivos, ya que una interfaz superior que tuviese que cumplir esta función complicaría el uso del sistema y la velocidad con la que este podría ejecutar sus acciones normales, es importante remarcar el hecho de que el sistema maneja muchas de estas funciones de forma análoga entre los distintos sistemas operativos, esto nos hace recordar la época en la cual los sistemas operativos se dividieron en sus dos principales vertientes, las cuales perduran hasta el día de hoy, sin embargo, debido principalmente a que las computadoras parten de una misma base, estructura y arquitectura es posible encontrarse con las mismas llamadas al sistema o al menos con llamadas al sistema equivalentes debido principalmente a estas similitudes entre los sistemas.

En conclusion podemos ver que hay unos formatos como el .txt, que pueden abrirse en los dos diferentes sistemas operativos, pero de la misma manera hay unos archivos que solo pueden ser abiertos por solo un formato de sistemas operativos. Podemos poner como ejemplo el archivo creado en word, el cual solo puede ser abierto y modificado en el sistema operativo de Windows, pero este formato no puede ser abierto o modificado en Linux.

En esta práctica, tuve la oportunidad de manejar Linux y Windows, dos de los sistemas operativos más utilizados. A través del uso de la interfaz de llamadas al sistema en lenguaje C, obteniendo

conceptos clave y habilidades que son esenciales para trabajar con estos sistemas.