

INSTITUTO POLITÉCNICO NACIONAL

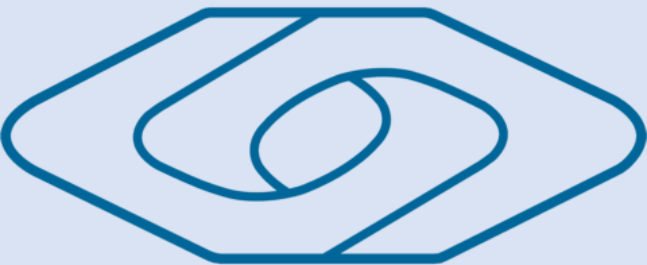
ESCUELA SUPERIOR DE COMPUTO



“Practica 1”

Sistemas Operativos

INSTITUTO POLITÉCNICO NACIONAL



ESCOM

- Chávez Rodríguez Héctor
 - Delgado Mendoza María Fernanda
 - Gómez Marbán Jonathan Said
 - Ramírez Farías Luis Antonio
-
- **Grupo:** 4CM1
-
- **Profesor:** Jorge Cortes Galicia

I. Introducción Teórica

¿Qué es un Sistema Operativo?

El sistema operativo es el conjunto de programas informáticos, que permite la administración eficaz de los recursos de un ordenador.

El sistema operativo también es conocido como sistema o software y puede definirse como el conjunto de programas que están hechos, específicamente, para ejecutar varias tareas en las que actúa como intermediario entre el usuario y el ordenador.

El sistema operativo representa el programa más importante de la computadora, ya que comienza a trabajar nada más encender el equipo, ya que se encarga de gestionar el hardware y permite la interacción con el usuario.

Algunas de las funciones básicas de este software son:

- Gestionar procesos o recursos para que los programas puedan ejecutarse de manera correcta.
- Administrar los puertos de entrada y salida, por ejemplo: micrófonos, altavoces, impresoras o el monitor.
- Garantizar la seguridad del ordenador, impidiendo el acceso a ciertos archivos o programas para el correcto funcionamiento del equipo.
- Administrar la memoria principal del dispositivo, de modo que, aunque varios programas se pongan en marcha, cada uno cuente con una entrada de memoria independiente.
- Detectar errores, mantener la operatividad y controlar dispositivos, de manera que se eviten las interrupciones.

Dos de los Sistemas Operativos más importantes son Windows Y Linux.

Windows es un sistema operativo creado por Microsoft. Consiste en un conjunto de programas que permiten la ejecución de los recursos que tiene un ordenador. El significado del término (Windows, ventanas) hace alusión a su interfaz gráfica, que presenta un modelo basado en tareas y compartimentos independientes,



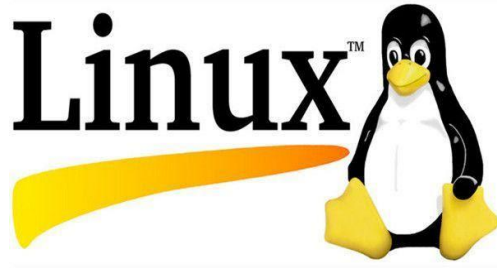
con sus propios menús y controles.

El año 1975 fue el inicio del fenómeno pronto responsable de la expansión del ordenador para uso personal a nivel mundial.

Algunas de sus versiones desde su comienzo son: Windows 95, Windows 98, Windows 2000, Windows XP, Windows 7, Windows 8 y Windows 10.

Por otro lado, Linux es un sistema operativo semejante a Unix, de código abierto y desarrollado por una comunidad, para computadoras, servidores, mainframes, dispositivos móviles y dispositivos embebidos. Es

compatible con casi todas las principales plataformas informáticas, incluyendo x86, ARM y SPARC, por lo que



es uno de los sistemas operativos más soportados.

Linus Torvalds lanzó el kernel de Linux en septiembre de 1991.

Cada versión del sistema operativo Linux gestiona los recursos de hardware, lanza y gestiona las aplicaciones, y proporciona alguna forma de interfaz de usuario. La enorme comunidad de desarrollo y la amplia gama de distribuciones significa que una versión de Linux está disponible para casi cualquier tarea, y Linux ha penetrado en muchas áreas de la informática.

II. Desarrollo Experimental

Linux

Distribuciones de Linux:

Una distribución Linux o también conocida coloquialmente como distro, es una distribución de software basada en el núcleo Linux y que incluye determinados paquetes para ofrecer ciertas características o funciones al usuario. Algunas de las distribuciones principales de Linux son:

-DEBIAN: Es una de las más antiguas y es muy estable. Si creas una distribución basada en Debian podrás contar con todas las actualizaciones de seguridad, rendimiento y compatibilidad que trae el sistema.

-UBUNTU: Nace con el objetivo de llevar Linux a todo el mundo, ya que anteriormente Linux era un sistema operativo difícil de manejar y requería conocimientos técnicos. Es muy usada en escritorio y servidores

-LINUX MINT: Está basada en Ubuntu y está orientada a un público principiante. De este modo, esta distro busca acercar Linux aún más a la comunidad general brindando una experiencia parecida a Windows. Si usas esta distro podrás ver una interfaz muy similar a la de Windows.

-RED HAT: Entre sus servicios brinda soluciones personalizadas a través de un gran soporte. Por lo que es la distribución más importante para el entorno corporativo.

-FEDORA: Es una distro basada en Red Hat para el usuario final. Usa el mismo manejador de paquetes y es gratuita. Si bien Fedora está desarrollada por una comunidad libre, Red Hat auspicia esta distro y contribuye a las mejoras del kernel.

-CentOs: Está basada en Red Hat, es gratuita y está destinada a la comunidad. Se hizo popular en los servidores porque brindaba la estabilidad y confiabilidad de Red Hat, pero sin tener que pagar.

-SUSE: Suse Enterprise Linux es una de las distribuciones más importantes y más antiguas de Linux. Así como Red Hat, esta distribución está orientada al mundo corporativo. También ofrece una distro para el usuario final llamada Open Suse.

¿Qué diferencia una distribución GNU/Linux de las demás?

Los paquetes y sus gestores

Si bien un mismo programa puede estar disponible para miles de distribuciones, el paquete con el cual se instala en cada una de ellas puede ser distinto. Cuando hablamos de paquete nos referimos al formato o extensión del archivo, el cual es usado por el sistema operativo para instalar un programa.

Algunos formatos de paquetes son:

- deb: usado por Debian y sus derivados.
- RPM: (Red Hat Package Manager) originario de Red Hat y usado extensivamente por muchas otras como Fedora, OpenSUSE, Mandriva, Mageia y otros.
- PISI: de Pardus.
- MO: de Slax.
- PUP y PET: Puppy Linux.
- .txz: Slackware

Y algunos de los gestores de paquetes más conocidos:

- APT (terminal) y Synaptic (gráfico): usado por Debian y sus derivados.
- Zypper (terminal) y YaST (gráfico): el gestor de OpenSUSE.
- YUM: Fedora y Yellow Dog Linux.
- Pacman: Arch Linux.
- dpkg: creado originalmente por Debian.
- Urpmi: Mandriva y Mageia.
- up2date: Red Hat.
- slapt-get, slackpkg y swaret: diversas “herramientas” que usa Slackware para trabajar con paquetes tgz.

La usabilidad

Otro tópico que se repite muy de seguido es el nivel de experiencia que precisamos tener para usar una distribución. Sucede muchas veces al aconsejar distribuciones a los novatos, o como nos habrá sucedido con nuestra primera distribución, que se suele escuchar “ni sueñes en probar Gentoo de entrada” o “Ubuntu es una buena opción para empezar”.

La dificultad de uso de una distribución se estima en cuanto a:

- la cantidad de elementos gráficos que ofrece.
- la cantidad de trabajo que obligatoriamente debe hacerse por consola (donde no existan alternativas gráficas para esa tarea).
- la dificultad de la instalación.
- el volumen de configuración que debe hacerse luego de la instalación de la distribución.
- si durante la instalación es necesario configurar la partición de discos o esta puede hacerse automáticamente.

Es por ello que es común agrupar ciertas distribuciones como las de nivel experto (Gentoo, Linux From Scratch, Slackware, Arch), las cuales no son recomendables para el usuario “intermedio-novato”.

El hardware

No es una de las primeras cosas que se menciona al hablar de una distribución, pero no deja de ser algo importante. En un mundo que trata de desligarse del “sistema que cada vez pide más recursos” (Windows) y aún así mantenerse al día con los nuevos hardwares, existe un nicho de distribuciones que permite reciclar el hardware (Puppy Linux, Slitaz, Tiny Core Linux, AUSTRUMI, Slax, Lubuntu, Xubuntu, Alcolix, Damn Small Linux, Molinux, etc.). Si bien otras distros, como ser Linux Mint o Arch pueden instalarse en computadoras antiguas, llega un límite en la cual la fluidez del sistema se pierde, por ello existen distribuciones especializadas para ese tipo de hardware. Es más lógico, por ende, que algunas de estas distribuciones ofrezcan soporte a versiones de 32 bits y 16 bits; las más difundidas ofrecen soporte 32 y 64 bits.

Los formatos de lanzamiento

Es simple: una distribución suele venir en una serie de formatos que terminan por definir la esencia de la misma. Si bien los Live CD/DVD suelen ser comunes entre las distros populares, existen muchas otras que no usan dicho formato, absteniéndose a lanzar solamente versiones instalables.

La posibilidad de contar con un CD, DVD, Live CD/DVD, distintos entornos de escritorio por defecto o capacidad para instalar desde internet, es algo que influye en la decisión de muchos usuarios para probar una distribución o usarla definitivamente. También vemos que existen pre-lanzamientos que permiten a la comunidad testear la distribución antes de estar finalizada.

Otros puntos importantes también incluyen aquellas versiones para dispositivos portátiles y otras que distribuyen “Spin-Offs”, donde el ejemplo más concreto es el de Fedora, que cuenta con versión para Juegos, de Laboratorio y de Diseño, si bien esto, a mi parecer, es cuestión de instalar paquetes existentes en distintos repositorios. Por último, no olvido las distribuciones “rolling-release”, cuyos exponentes más claros son [Debian](#), [Arch](#) y [OpenSUSE](#), permitiendo actualizar el software y las versiones del sistema sin necesidad de realizar una nueva instalación o el miedo a perder datos personales.

El objetivo general

Cada distribución tiene en mente un objetivo con el cual pretende llegar a sus usuarios actuales o potenciales. De ahí podemos diferenciar a las que son específicas para portátiles (como JoliCloud que también está orientada al uso en la nube) y las de servidores ([Red Hat Linux Enterprise](#) es una de las más robustas y con mejor soporte actualmente).

Otras distribuciones tienen como objetivo el cuidado estético del escritorio y la similitud con otros sistemas (facilitando así la transición con estos), como es el caso de [PearOS](#) (con una estética similar a Mac), [ZorinOS](#) (que adapta GNOME para ofrecer un entorno similar a distintas versiones de Windows) y [ElementaryOS](#) (con un juego de íconos integrados y una instalación funcional por defecto); estas pueden presumir de diferenciarse de los entornos clásicos, pero estos aún pueden ser instalados en dichas distribuciones.

COMANDOS

-Comando **ls**: Nos permite listar el contenido de un directorio.

```
sysadmin@localhost:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
```

-Comando **cd**: Se utiliza para navegar por los archivos y directorios.

```
sysadmin@localhost:~$ cd ~/Documents
```

-Comando **cat**: Significa concatenar, se usa mayormente para visualizar la información de cualquier archivo rápidamente

```
sysadmin@localhost:~/Documents$ cat animals.txt
1 retriever
2 badger
3 bat
4 wolf
5 eagle
```

-Comando **ps**: Muestra por pantalla un listado de los procesos que se están ejecutando en el sistema

```
sysadmin@localhost:~$ ps
  PID TTY          TIME CMD
   80 pts/0        00:00:00 bash
   94 pts/0        00:00:00 ps
```

-Comando **grep**: Es un filtro de texto que busca líneas de entrada y devolverá aquellas que coincidan con un patrón determinado. También se puede utilizar para filtrar y obtener información sobre un usuario específico.

```
sysadmin@localhost:~/Documents$ grep sysadmin passwd
sysadmin:x:1001:1001:System Administrator,,,:/home/sysadmin:/bin/bash
```

-Comando **vi**: Es el principal editor de texto dentro de Linux. Simplemente se escribe el comando, seguido de la ruta del archivo que se quiere editar o crear.

```
sysadmin@localhost:~$ vi newfile.txt
```

-Comando **chmod**: Permite cambiar los permisos de acceso de un fichero o directorio. Aquí sysadmin es el propietario del archivo, u representa el conjunto de permisos del propietario, y el + para indicar que se añade un permiso y el x para representar el permiso de ejecución.

```
sysadmin@localhost:~/Documents$ chmod u+x hello.sh
```

-Comando **ls**: lista los archivos y directorios de la ruta en la que nos encontremos

```
jona@jona-VirtualBox:~$ ls
Descargas  Escritorio  Música      Público
Documentos Imágenes    Plantillas  Vídeos
```

-Comando **ls -l**: lista los archivos y directorios de la ruta en la que nos encontremos mostrando detalles de los archivos, como el tipo de archivo, los permisos, el usuario propietario, la marca de tiempo y la cantidad de enlaces duros.

```
jona@jona-VirtualBox:~$ ls -l
total 32
drwxr-xr-x 2 jona jona 4096 ago 24 01:29 Descargas
drwxr-xr-x 2 jona jona 4096 ago 24 17:09 Documentos
drwxr-xr-x 2 jona jona 4096 ago 24 01:29 Escritorio
drwxr-xr-x 2 jona jona 4096 ago 24 01:29 Imágenes
drwxr-xr-x 2 jona jona 4096 ago 24 01:29 Música
drwxr-xr-x 2 jona jona 4096 ago 24 01:29 Plantillas
drwxr-xr-x 2 jona jona 4096 ago 24 01:29 Público
drwxr-xr-x 2 jona jona 4096 ago 24 01:29 Vídeos
```


-Comando **ls -la**: combinación de opciones del comando **ls** (**-l -a**), muestra una lista de ficheros detallada incluyendo los ficheros ocultos, en este caso se están agrupando las opciones.

```
jona@jona-VirtualBox:~$ ls -la
total 96
drwxr-xr-x 16 jona jona 4096 ago 25 10:53 .
drwxr-xr-x  3 root root 4096 ago 24 00:55 ..
-rw-r----- 1 jona jona  165 ago 24 11:19 .bash_history
-rw-r----- 1 jona jona  220 ago 24 00:55 .bash_logout
-rw-r----- 1 jona jona 3771 ago 24 00:55 .bashrc
drwx----- 13 jona jona 4096 ago 24 11:40 .cache
drwxr-xr-x 14 jona jona 4096 ago 24 02:16 .config
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Descargas
drwxr-xr-x  2 jona jona 4096 ago 24 17:09 Documentos
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Escritorio
drwx-----  3 jona jona 4096 ago 25 10:53 .gnupg
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Imágenes
drwx-----  3 jona jona 4096 ago 24 01:29 .local
drwx-----  5 jona jona 4096 ago 24 11:40 .mozilla
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Música
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Plantillas
-rw-r----- 1 jona jona   807 ago 24 00:55 .profile
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Público
drwx-----  2 jona jona 4096 ago 24 01:31 .ssh
-rw-r----- 1 jona jona    0 ago 24 01:32 .sudo_as_admin_successful
-rw-r----- 1 jona jona    5 ago 25 10:52 .vboxclient-clipboard.pid
-rw-r----- 1 jona jona    5 ago 25 10:52 .vboxclient-display-svgx-x11.pid
-rw-r----- 1 jona jona    5 ago 25 10:52 .vboxclient-draganddrop.pid
-rw-r----- 1 jona jona    5 ago 25 10:52 .vboxclient-seamless.pid
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Videos
```

-Comando **pwd**: de las siglas en inglés print working directory, se utiliza para imprimir el nombre del directorio actual.

```
jona@jona-VirtualBox:~$ pwd
/home/jona
```

-Comando **clear**: se usa para limpiar la pantalla de la terminal

```
-rw-r----- 1 jona jona    5 ago 25 10:52 .vboxclient-seamless.pid
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Videos
jona@jona-VirtualBox:~$ clear
jona@jona-VirtualBox:~$
```

-Comando **pwd**: de las siglas en inglés print working directory, se utiliza para imprimir el nombre del directorio actual.

```
jona@jona-VirtualBox:~$ pwd
/home/jona
```

-Comando **ls -la | more**:

Comando **more**: en solitario se usa para ver los archivos de texto en el símbolo del sistema, mostrando una pantalla a la vez en caso de que el archivo sea grande. También permite al usuario desplazarse hacia arriba y hacia abajo por la página

Símbolo '|': Una tubería es una secuencia de una o más órdenes separadas por el carácter '|'. En este caso la salida estándar de una orden se conecta con la entrada estándar de la siguiente orden

La combinación del comando **ls -la** y el comando **more** conectados por '|', muestra una lista de ficheros detallada incluyendo los ficheros ocultos

```
jona@jona-VirtualBox:~$ ls -la | more
total 96
drwxr-xr-x 16 jona jona 4096 ago 25 10:53 .
drwxr-xr-x  3 root root 4096 ago 24 00:55 ..
-rw-r--r--  1 jona jona  165 ago 24 11:19 .bash_history
-rw-r--r--  1 jona jona  220 ago 24 00:55 .bash_logout
-rw-r--r--  1 jona jona 3771 ago 24 00:55 .bashrc
drwx----- 13 jona jona 4096 ago 24 11:40 .cache
drwxr-xr-x 14 jona jona 4096 ago 24 02:16 .config
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Descargas
drwxr-xr-x  2 jona jona 4096 ago 24 17:09 Documentos
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Escritorio
drwx-----  3 jona jona 4096 ago 25 10:53 .gnupg
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Imágenes
drwx-----  3 jona jona 4096 ago 24 01:29 .local
drwx-----  5 jona jona 4096 ago 24 11:40 .mozilla
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Música
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Plantillas
-rw-r--r--  1 jona jona  807 ago 24 00:55 .profile
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Público
drwx-----  2 jona jona 4096 ago 24 01:31 .ssh
-rw-r--r--  1 jona jona    0 ago 24 01:32 .sudo_as_admin_successful
-rw-r-----  1 jona jona    5 ago 25 10:52 .vboxclient-clipboard.pid
-rw-r-----  1 jona jona    5 ago 25 10:52 .vboxclient-display-svga-x11.pid
-rw-r-----  1 jona jona    5 ago 25 10:52 .vboxclient-draganddrop.pid
-rw-r-----  1 jona jona    5 ago 25 10:52 .vboxclient-seamless.pid
drwxr-xr-x  2 jona jona 4096 ago 24 01:29 Vídeos
```

COMANDO LINUX: CP

El comando linux cp es uno de los comandos básicos que cualquier persona interesada en aprender Linux debe saber.

Este comando sirve para copiar archivos y directorios dentro del sistema de archivos (file system) de un sistema Linux.

Utilizar este comando es muy simple pues solo indicas el nombre del archivo a copiar y el nombre del archivo o directorio donde quieres que se copie, pero también tiene algunas otras funciones mediante algunos parámetros que te pueden ser útiles en un momento dado.

Sintaxis del comando linux cp

```
cp [opciones]... [-T] origen destino
cp [opciones]... origen... directorio
cp [opciones]... -t directorio origen...
```

Estas son algunas de las opciones que tiene el comando linux cp:

OPCIÓN	DESCRIPCIÓN
-a	archive files
-f	copia forzada al remover el archivo destino si es necesario
-i	interactivo, pide confirmación del usuario antes de sobre escribir
-l	enlaces en vez de copia
-L	seguir enlaces simbólicos
-n	no sobre escribir archivos
-R	copia recursiva recursive copy (including hidden files)
-u	actualizar, copia cuando la fuente es más reciente que el destino
-v	muestra mensajes informativos

COMANDO LINUX: MV

MV es un comando de Unix usado para mover o renombrar archivos o directorios del sistema de archivos.

El archivo original es borrado y se crea un nuevo archivo con el mismo contenido, el nombre puede ser diferente o puede ser el mismo.

La sintaxis para utilizar el comando linux mv es la siguiente:

```
mv [OPCION]... [-T] ORIGEN DESTINO
mv [OPCION]... ORIGEN... DIRECTORIO
mv [OPCION]... -t DIRECTORIO ORIGEN...
```

1.- Cambiar el nombre de un archivo

Este es el uso más común de mv pero que a un usuario principiante le puede parecer extraño. Por lo general la primer idea que se viene a la cabeza cuando ves mv es que sirve para mover archivos y no para renombrar archivos. Esta pequeña confusión ocurre sobre todo si vienes de sistemas como Windows, pero el comando linux mv es el comando adecuado cuando necesitas que un archivo o directorio tenga un nombre diferente.

```
$ mv nombre1.txt archivo1.txt
```

En el ejemplo anterior el nombre del archivo «nombre1.txt» es cambiado por el nombre «archivo1.txt», de la forma que se encuentra el archivo solo cambiará de nombre y no de ubicación.

2.- Re nombrar un directorio

Cambiar el nombre de un directorio es prácticamente lo mismo que un archivo, recuerda que en Linux «Todo es un Archivo» por lo que usar el comando mv con un directorio para cambiar su nombre es igual.

```
$ mv directorio nueva-carpeta
```

En este ejemplo «directorio» es un folder o carpeta que cambia de nombre para llamarse «nueva-carpeta» al igual que el ejemplo anterior no se cambia de ubicación, sin embargo para que esto ocurra no debe existir el directorio destino.

3.- Mover archivos de ubicación

Para mover un archivo a una ubicación diferente el DESTINO debes ser un directorio existente, si este es el caso el archivo mantendrá el mismo nombre pero se trasladará a la nueva ruta.

```
$ mv archivo1.txt ./nueva-carpeta
```

Si tu ves el contenido del directorio «nueva-carpeta» con el [comando ls](#), podrás apreciar ahora que «archivo1.txt» se encuentra dentro de «nueva-carpeta». Primero mira el contenido de la carpeta actual, donde encontrarás un archivo y un directorio.

```
$ ls -l -rw-r--r-- 1 lm lm 0 Agost 25 17:37 archivo1.txt drwxr-xr-x 2 lm  
lm 4096 Agost 25 17:37 nueva-carpeta
```

Al ejecutar el comando mv y luego al volver a listar los archivos de forma recursiva podemos ver que en la carpeta actual solo esta el directorio «nueva-carpeta» y ahora dentro de esta se encuentra «archivo1.txt», mira el ejemplo:

```
$ mv archivo1.txt nueva-carpeta $ ls -lR .: drwxr-xr-x 2 lm lm 4096 Agost  
25 17:37 nueva-carpeta  
./nueva-carpeta: total 0 -rw-r--r-- 1 lm lm 0 Agost 25 17:37 archivo1.txt
```

COMANDO LINUX: MKDIR

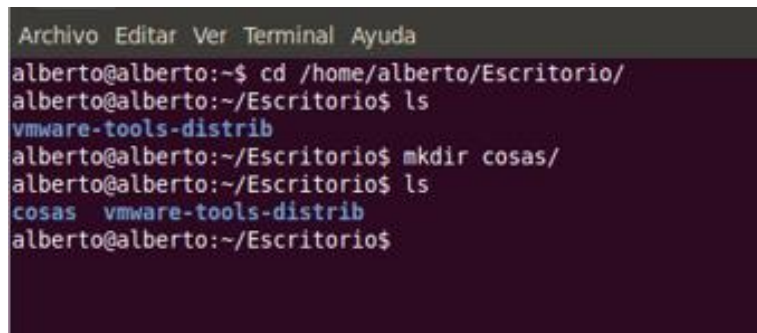
El comando mkdir es un programa o instrucción disponible en los sistemas operativos del tipo UNIX (como GNU/Linux), DOS, OS/2 y Windows.

Esta herramienta es usada para crear un nuevo subdirectorio o carpeta del sistema de archivos

Como funciona:

Una vez en el terminal, escribir `mkdir nombre del directorio a crear/`, y se creara el directorio donde este

Por ejemplo si estáis en `/home/user1/Escritorio` y haceís `mkdir cosas/`, en el escritorio saldrá el directorio cosas (una carpeta que se llama cosas).



```
Archivo Editar Ver Terminal Ayuda
alberto@alberto:~$ cd /home/alberto/Escritorio/
alberto@alberto:~/Escritorio$ ls
vmware-tools-distrib
alberto@alberto:~/Escritorio$ mkdir cosas/
alberto@alberto:~/Escritorio$ ls
cosas  vmware-tools-distrib
alberto@alberto:~/Escritorio$
```

EL COMANDO LINUX RMDIR

El comando linux rmdir sirve para borrar directorios, este comando es muy simple y no tiene muchos parámetros. En general se puede decir que rmdir es útil cuando tienes que borrar directorios que están vacíos, en otro caso es mejor utilizar otro comando.

Para usar este comando solo tienes que indicar el directorio que deseas borrar. Por ejemplo:

`rmdir temporal`

Este comando borraría el directorio con nombre «temporal» y será borrado si el directorio está vacío. En caso contrario te mostrará un mensaje indicando que ese directorio contiene archivos o directorios por lo que no puede ser borrado.

Si quieres borrar una serie de directorios vacíos que están unos dentro de otros, por ejemplo `./a/b/d`, `./a/b/c` y `./a/o` estos directorios tendrían una estructura como esta:

```
-- a | +-- b | +-- c | +-- d | +-- o
```

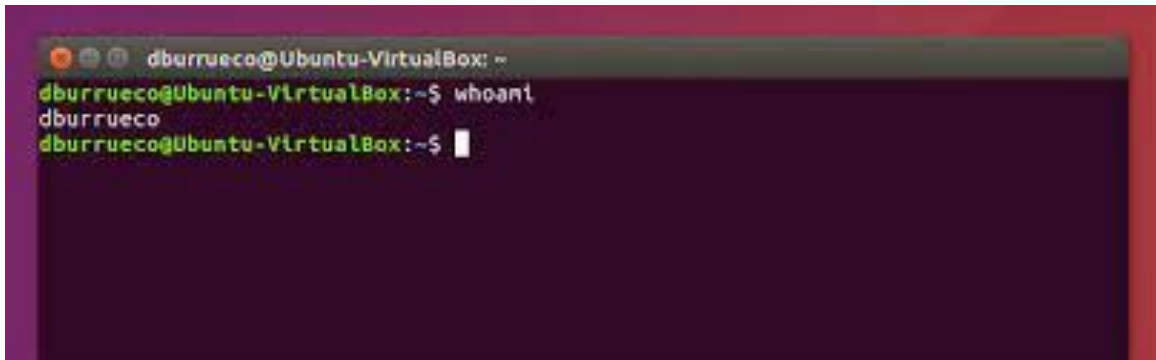
Para borrar estos directorios puedes utilizar el comando linux rmdir con el parámetro `-p` que borra todos los directorios que estén dentro de la ruta indicada, así para borrar el directorio `a` y todos los directorios vacíos dentro de este, se utilizaría el siguiente comando:

`rmdir -p a`

COMANDO LINUX: WHOAMI

El comando `whoami` es un comando del tipo Unix, proviene de la concatenación de las palabras en inglés ¿Who am I? que significa, ¿Quién soy?.

Es un comando simple, utilizado para imprimir el nombre de usuario efectivo del usuario actual cuando se invoca, que se entiende como el nombre de el usuario en sesión.

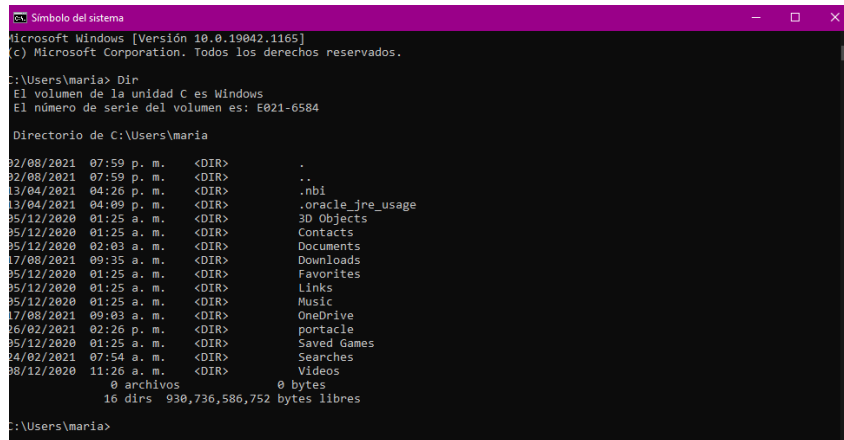
A screenshot of a terminal window titled 'dburruco@Ubuntu-VirtualBox: ~'. The prompt is 'dburruco@Ubuntu-VirtualBox:~\$'. The user has entered the command 'whoami'. The output is 'dburruco'. The prompt is now 'dburruco@Ubuntu-VirtualBox:~\$' with a cursor at the end.

```
dburruco@Ubuntu-VirtualBox: ~  
dburruco@Ubuntu-VirtualBox:~$ whoami  
dburruco  
dburruco@Ubuntu-VirtualBox:~$
```

Windows

Comandos

-Comando **Dir**: permitir realizar búsquedas de archivos y directorios



```
Símbolo del sistema
Microsoft Windows [Versión 10.0.19042.1165]
(c) Microsoft Corporation. Todos los derechos reservados.

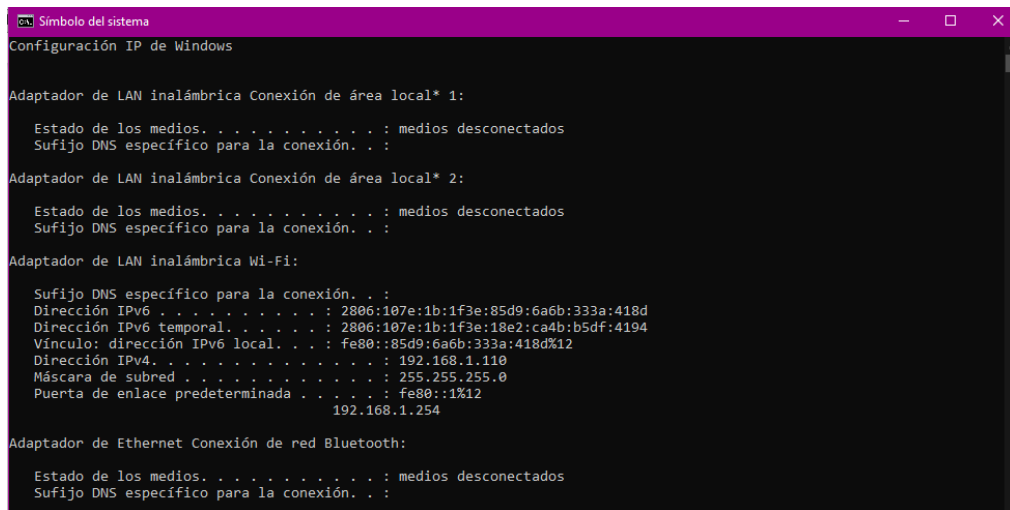
C:\Users\maria> Dir
El volumen de la unidad C es Windows
El número de serie del volumen es: E821-6584

Directorio de C:\Users\maria

02/08/2021  07:59 p. m.      <DIR>          .
02/08/2021  07:59 p. m.      <DIR>          ..
13/04/2021  04:26 p. m.      <DIR>          .nbi
13/04/2021  04:09 p. m.      <DIR>          .oracle_jre_usage
05/12/2020  01:25 a. m.      <DIR>          3D Objects
05/12/2020  01:25 a. m.      <DIR>          Contacts
05/12/2020  02:03 a. m.      <DIR>          Documents
17/08/2021  09:35 a. m.      <DIR>          Downloads
05/12/2020  01:25 a. m.      <DIR>          Favorites
05/12/2020  01:25 a. m.      <DIR>          Links
05/12/2020  01:25 a. m.      <DIR>          Music
17/08/2021  09:03 a. m.      <DIR>          OneDrive
06/02/2021  02:26 p. m.      <DIR>          portacle
05/12/2020  01:25 a. m.      <DIR>          Saved Games
14/02/2021  07:54 a. m.      <DIR>          Searches
06/12/2020  11:26 a. m.      <DIR>          Videos
               0 archivos             0 bytes
               16 dirs  930,736,586,752 bytes libres

C:\Users\maria>
```

Comando **ipconfig**: Además de darle la dirección IP de la computadora actual, también le brinda la dirección IP de tu enrutador (router), su dirección MAC y le permite eliminar tu DNS.



```
Símbolo del sistema
Configuración IP de Windows

Adaptador de LAN inalámbrica Conexión de área local* 1:

Estado de los medios. . . . . : medios desconectados
Sufijo DNS específico para la conexión. . :

Adaptador de LAN inalámbrica Conexión de área local* 2:

Estado de los medios. . . . . : medios desconectados
Sufijo DNS específico para la conexión. . :

Adaptador de LAN inalámbrica Wi-Fi:

Sufijo DNS específico para la conexión. . :
Dirección IPv6 . . . . . : 2806:107e:1b:1f3e:85d9:6a6b:333a:418d
Dirección IPv6 temporal. . . . . : 2806:107e:1b:1f3e:18e2:ca4b:b5df:4194
Vínculo: dirección IPv6 local. . . : fe80::85d9:6a6b:333a:418d%12
Dirección IPv4. . . . . : 192.168.1.110
Máscara de subred . . . . . : 255.255.255.0
Puerta de enlace predeterminada . . . : fe80::1%12
                                   192.168.1.254

Adaptador de Ethernet Conexión de red Bluetooth:

Estado de los medios. . . . . : medios desconectados
Sufijo DNS específico para la conexión. . :
```



```

C:\Users\maria>ipconfig

Configuración IP de Windows

Adaptador de LAN inalámbrica Conexión de área local* 1:

    Estado de los medios. . . . . : medios desconectados
    Sufijo DNS específico para la conexión. . . :

Adaptador de LAN inalámbrica Conexión de área local* 2:

    Estado de los medios. . . . . : medios desconectados
    Sufijo DNS específico para la conexión. . . :

Adaptador de LAN inalámbrica Wi-Fi:

    Sufijo DNS específico para la conexión. . . :
    Dirección IPv6 . . . . . : 2806:107e:1b:1f3e:85d9:6a6b:333a:418d
    Dirección IPv6 temporal. . . . . : 2806:107e:1b:1f3e:18e2:ca4b:b5df:4194
    Vínculo: dirección IPv6 local. . . : fe80::85d9:6a6b:333a:418d%12
    Dirección IPv4. . . . . : 192.168.1.110
    Máscara de subred . . . . . : 255.255.255.0
    Puerta de enlace predeterminada . . . . : fe80::1%12
                                                192.168.1.254

Adaptador de Ethernet Conexión de red Bluetooth:

    Estado de los medios. . . . . : medios desconectados
    Sufijo DNS específico para la conexión. . . :

C:\Users\maria>

```

-Comando **Cls**: Borra la pantalla, es decir, borra la ventana del Símbolo del sistema

```

C:\Users\maria>cls

C:\Users\maria>

```

-Comando **Ver**: Muestra el número de versión del sistema operativo

```

C:\Users\maria>ver

Microsoft Windows [Versión 10.0.19042.1165]

C:\Users\maria>

```

-Comando **Tree**: Permitirá sacar un listado de la estructura de carpetas de una unidad de disco o ruta

```

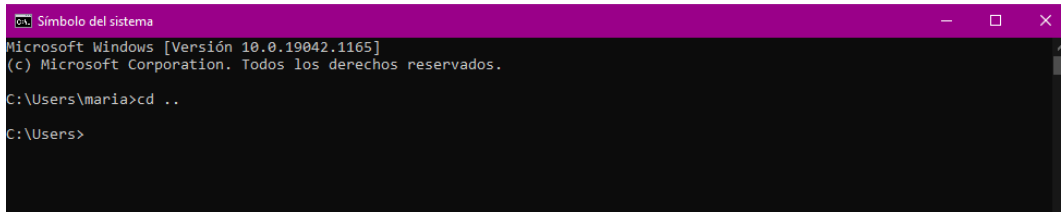
C:\Users\maria>tree

Listado de rutas de carpetas para el volumen Windows
El número de serie del volumen es E021-6584
C:.
.
..
.nbi
--downloads
--log
--product-cache
--jdk
--18.0.111.1464
--nb-base
--8.2.0.0.201609300101
--nb-extide
--8.2.0.0.201609300101
--nb-javase
--8.2.0.0.201609300101
--tmp
--wd
--oracle_jre_usage
--3D Objects
--Contacts
--Documents
--Downloads
--Favorites
--Lenovo
--Links
--Music
--OneDrive
--documentos
--Blocs de notas de OneNote
--NetBeansProjects
--Agregación
--build
--classes
--agregación
--nbproject
--private
--src
--agregación
--test
--banco
--build

```

especifica

Comando cd: Sirve para cambiar de directorio



```
Símbolo del sistema
Microsoft Windows [Versión 10.0.19042.1165]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\maria>cd ..

C:\Users>
```

-Comando type: muestra el contenido de un archivo de texto.

```
C:\Users\jona->type Ejemplo.txt
Esto es un ejemplo.
```

-Comando mkdir: Crea un directorio

```
C:\Users\jona->mkdir ejemplo
C:\Users\jona->cd ejemplo
C:\Users\jona-\ejemplo>
```

-Comando rmdir: Elimina un directorio

```
C:\Users\jona->rmdir ejemplo
C:\Users\jona->cd ejemplo
El sistema no puede encontrar la ruta especificada.
```

-Comando del: Elimina un archivo

```
C:\Users\jona->del Ejemplo.txt
```

-Comando copy: Permite copiar un archivo de texto en una ubicación alternativa.

```
C:\Users\jona->copy GomezMarban_tarea1.docx ejemploCopiar.docx
1 archivo(s) copiado(s).
```

```
C:\Users>copy C:\users\jona-\GomezMarban_tarea1.docx C:\users\jona-\documents\nuevoejemploCopiar.docx
1 archivo(s) copiado(s).
```

-Comando ren: Permite cambiar el nombre a un archivo.

```
C:\Users\jona->type Ejemplo.txt
Esto es un ejemplo.
C:\Users\jona->ren Ejemplo.txt ejemploEquipo7.txt

C:\Users\jona->type ejemploEquipo7.txt
Esto es un ejemplo.
C:\Users\jona->type Ejemplo.txt
El sistema no puede encontrar el archivo especificado.
```

-Comando **chdir**: Permite cambiar el directorio. Realiza la misma acción que el comando **cd**.

```
C:\Users\jona->chdir documents
C:\Users\jona-\Documents>
```

-Comando **echo**: Permite mostrar mensajes

```
C:\Users\jona->echo esto es un ejemplo
esto es un ejemplo
```

Pero, si se usa sin parámetros, **echo** muestra la configuración de **eco** actual:

```
C:\Users\jona->echo
ECHO está activado.
```

-Comando **find**: Se utiliza para buscar una cadena de texto específica en un archivo

```
C:\Users\jona->find "es" ejemploEquipo7.txt
----- EJEMPLOEQUIPO7.TXT
Esto es un ejemplo.

C:\Users\jona->find "no" ejemploEquipo7.txt
----- EJEMPLOEQUIPO7.TXT
```

Si los caracteres especificados a buscar se encuentran en el archivo, entonces, muestra la línea o líneas donde coincide la subsecuencia. En caso contrario, no muestra nada.

Programas

Figura

Programa basado en ciclos anidados, esto para la creacion solicitada, teniendo la siguiente solución para este.

Funcion main.

- Solicitamos el numero de a diagonal solicitada
- Guardamos el numero
- Damos dos saltos de linea
- Declaramos variables enteras
- n2 guardara la division del numero ingresado entre 2
- n3 sera la multiplicacion de esta division por 2
- Guardamos este valor en n4
- Invocamos la funcion superior
- Generamos un salto de linea
- Invocamos la funcion inferior

```
int main(){
    int n;

    printf("Ingresa el numero de la diagonal\n");
    scanf("%i", &n);
    printf("%i\n\n", n);
    int n2,n3,n4;

    n2= n/2;
    n3= 2*n2;
    n4 = n2;

    superior(n, n2, n3,1);
    printf("\n");
    inferior(n, n2, 2,n2);

    return 0;
}
```

Función Superior.

Esta funcio sera la encargada de realizar la parte superior de la figura mediante los siguientes pasos:

- Condicion que si 2 es igual a 0.
- Ciclo que imprimira la ultima fila.
- de lo contrario, imprimira el primer espacio de la figura
- Ciclo que imprimira el lado izquierdo
- ciclo que genera el espacio entre la primer figura
- ciclo que imprimira el lado derecho
- ciclo que genera el espacio entre el rombo y la segunda figura
- ciclo para generar la ultima figura
- ciclo para generar el ultimo espacio
- salto de linea entre cada fila
- Al ser superior, ira descontando dos asteriscos e incrementara la altura

```
void superior(int na, int ne, int ns,int ni){
    int i,j,k,l,r,m;

    if(ns == 0){
        for(r = 0; r<3;r++){
            for(i = 0; i<na; i++){
                printf("*");
            }
        }
    }
    else{
        for(i = 0; i< ne; i++){
            printf(" ");
        }
        for(j = 0 ; j < ni ; j++){
            printf("*");
        }for(k = 0; k < ns; k++){
            printf(" ");
        }
        for(m = ns; m < na; m++){
            printf("*");
        }for(k = 0; k < ns; k++){
            printf(" ");
        }
        for(j = 0 ; j < ni ; j++){
            printf("*");
        }for(k = 0; k < ns; k++){
            printf(" ");
        }
        printf("\n");
        superior(na,ne,ns-2,ni+1);
    }
}
```

Función Inferior

Mediante esta función será la forma invertida de la función anterior, ya que realizara la parte inferior de la figura. Con los siguientes pasos se genera la figura solicitada.

- Condicion para generar el primer espacio inferior
- Ciclo para generar el espacio entre margen y la figura
- ciclo para generar la el lado izquierdo inferior
- ciclo para generar el espacio de la primer figura con el rombo inferior
- ciclo para imprimir el rombo
- ciclo para generar el espacio del rombo a la segunda figura
- ciclo para generar la segunda figura
- genera el ultimo espacio
- salto de linea
- Se vuelve a invocar la figura sumando dos asteriscos y restando la altura

```
void inferior(int na, int ne, int ns, int ni){
    int i,j,k,l,r,m;

    if(ns == 2*(ne+1)){
        printf(" ");
    }
    else{
        for(i = 0; i < ne; i++){
            printf(" ");
        }
        for(j = 0; j < ni; j++){
            printf("*");
        }
        for(k = 0; k < ns; k++){
            printf(" ");
        }
        for(m = ns; m < na; m++){
            printf("*");
        }
        for(k = 0; k < ns; k++){
            printf(" ");
        }
        for(j = 0; j < ni; j++){
            printf("*");
        }
        for(k = 0; k < ns; k++){
            printf(" ");
        }
        printf("\n");
        inferior(na, ne, ns+2, ni-1);
    }
}
```

Fibonacci

Al ser un programa con pocas instrucciones, se realizara en el main. Se crean tres variables enteras, mediante un ciclo este realizara la operación de suma y acumular este resultado, imprimera el primer digito y en seguido el segundo.

```
int main(){
    int a = 1, b = 0, i;

    for(i = 0; i < 10; i++){
        printf("%d\n", b);
        printf("%d\n", a);
        b = b + a;
        a = a + b;
    }
    return 0;
}
```

Balanceo de Parentesis

Se inicializa una estructura, esto para la forma de almacenamiento de los datos a ingresar.

```
struct nodo{  
    char simbolo;  
    struct nodo *sig;  
};
```

Funcion Insertar

La primer funcion insertar tipo void con parámetro char, esta será la ecuacion a enviar, se inicializa una variable tipo nodo, a esta variable se reserva memoria (ya que sera acomodado de una forma PILA), a esta variable se le asignara una variable de la estructura guardando la cadena enviada, con esto mediante una condición dirá si la raíz esta vacia, a esta raiz se guardara la cadena y a siguiente se le asignara NULO, de lo contrario al siguiente se guardara la raiz y a raiz la cadena.

```
void insertar (char x){  
    struct nodo *nuevo;  
    nuevo = (struct nodo*) malloc(sizeof(struct nodo));  
    nuevo->simbolo=x;  
    if(raiz == NULL){  
        raiz=nuevo;  
        nuevo->sig=NULL;  
    }  
    else{  
        nuevo->sig=raiz;  
        raiz=nuevo;  
    }  
}
```

Funcion Extraer

Función con retorno carácter sin parámetros, si la raiz es diferente a NULO, se crea la variable que extraera la información, a la variable raiz se le asigna el valor siguiente, libera memoria y retorna el carácter, de lo contrario retorna -1.

```
char extraer (){  
    if(raiz != NULL){  
        char info= raiz->simbolo;  
        struct nodo *bor = raiz;  
        raiz=raiz->sig;  
        free(bor);  
        return info;  
    }  
    else{  
        return -1;  
    }  
}
```

Función liberar.

El objetivo de esta función es liberar todos los nodos de la PILA, esto cuando el programa termine de ejecutarse.

```
void liberar(){  
    struct nodo *reco =raiz;  
    struct nodo *bor;  
    while(reco != NULL){  
        bor=reco;  
        reco=reco->sig;  
        free(bor);  
    }  
}
```

Función vacia

Funcion que verifica si la PILA esta vacia, si esto es así retorna un 1, de lo contrario un 0.

```
int vacia(){
    if(raiz == NULL){
        return 1;
    }else{
        return 0;
    }
}
```

Funcion cargar

Está función tipo void con parámetro carácter, sera la que envíe reciba la formula y este la lleve al proceso necesario.

```
void cargar (char *formula){
    printf("\n\n\tIngrese la formula: ");
    gets(formula);
}
```

Funcion verificar

Funcion principal, ya que esta se encargara de verificar que la formula este balanceada, mediante un for ira realizando un recorrido (carácter por carácter), validando que el parentesis, corchete, llave cierre. Con ayuda de sentencias condicionales validara esto, la condicion expresa que mientras el carácter '(' ó '[' ó '{' este en la formula lo inserte, de lo contrario, si el carácter es ')', ']', '}' extraiga '(', '[', '{', de lo contrario la pila está vacia si no retornara un 0.

```
int verificar(char *formula){
    for(int f=0; f<strlen(formula);f++)
    {
        if(formula[f] == '(' || formula[f] == '[' || formula[f] == '{'){
            insertar(formula[f]);
        }
        else{
            if(formula[f] == ')'){
                if(extraer() != '('){
                    return 0;
                }
            }
            else
            {
                if(formula[f] == ']'){
                    if(extraer() != '['){
                        return 0;
                    }
                }
                else{
                    if(formula[f] == '}'){
                        if(extraer() != '{'){
                            return 0;
                        }
                    }
                }
            }
        }
    }
    if(vacia()){
        return 1;
    }else{
        return 0;
    }
}
```

Función menu

Esta función hará la "función del main", esto para no cargar la función principal, esta función tendrá un arreglo, el cual será el que guarde la fórmula ingresada, mostrará en pantalla el título "Parentesis Balanceados", mediante la función cargar enviaremos la fórmula, con una condicional verificará la fórmula, si esta cumple con lo necesario mostrará "La fórmula está balanceada", de lo contrario "La fórmula no está balanceada".

```
void menu(){
char formula[101];

    system("cls");
    system("color 0a");
    printf("\t\t\t\t----Parentesis Balanceados----");
    printf("\nEQUIPO 7");
    cargar(formula);
    if(verificar(formula)){
        printf("\n\tLa fórmula está balanceada\n\n");
    }
    else{
        printf("\n\tLa fórmula no está balanceada\n\n");
    }
}
```

Función main

Función principal, la cual donde el compilador se ejecutará, este mediante un ciclo realizará las suficientes repeticiones necesarias, esta solo invocará la función menu, en dado caso que ya no se desee ejecutar de nuevo, liberará la memoria y terminará de ejecutar.

```
main(){

    int opc;
    do
    {
        menu();
        system("pause");
    }while(opc != 2);
    liberar();
    return 0;
}
```

Expresiones

Función Introducir Expresión

Esta función pide al usuario que introduzca la expresión a evaluar.

```
void introducirExpresion(char *numero){
    printf("Introduce la expresion: ");
    fgets(numero, 256, stdin);
}
```

Función Identificar Operador

Función tipo entera con parámetro carácter, esta realiza que si es un paréntesis regresa un -1 y si es un operando un 0.

```
int identificarOperador(char operador){
    if (operador == '+'){
        return 1;
    }else if (operador == '-'){
        return 2;
    }else if (operador == '/'){
        return 3;
    }else if (operador == '*'){
        return 4;
    }else if (operador == '(' || operador == ')'){
        return -1;
    }else{
        return 0;
    }
}
```

Función Ejecutar Operación

Función tipo entera con tres parámetros enteros, esta función regresa el valor de alguna operación (suma, resta, multiplicación y división), identificando cual es el valor del operador.

```
int ejecutarOperacion(int numero1, int numero2, int operador){
    if (operador == 1){
        return numero1+numero2;
    }else if (operador == 2){
        return numero1-numero2;
    }else if (operador == 3){
        return numero1/numero2;
    }else if (operador == 4){
        return numero1*numero2;
    }else{
        return 0;
    }
}
```

Función Cuenta Dígitos

Función entera, regresa de cuantos dígitos es algún número, para ello requiere la cadena que contiene la expresión y la posición a partir de la que empieza el número.

```
int cuentaDigitos(char* numero, int posicion){
    int contador = 0;

    while (identificarOperador(numero[posicion]) == 0){
        posicion++;
        contador++;
    }
    return contador;
}
```


Función Convertir a Entero

Esta función convierte una cadena a entero, dicha cadena simboliza un numero y la convierte a tipo entero, para eso necesita saber de cuantos digitos es el numero, la cadena y posicion en la que se encuentra el numero. Regresa el numero que convirtio.

```
int convertirAEntero(int digitos, int posicion, char* numero){
    char nuevoNum[255];
    char* numeroF;
    int i;
    for (i = 0; i < digitos; i++){
        nuevoNum[i] = numero[posicion];
        posicion++;
    }

    int numeroR = atoi(nuevoNum);
    return numeroR;
}
```

- Aquí almacenamos solamente la cadena del numero.
- Esta función sirve para hacer un casting de tipo apuntador a caracter a entero.

Funcion main

Función principal, el cual tendra la tarea de lo siguiente:

- Solicitamos la cadena
- Se itera hasta que se termine de recorrer toda la cadena dada.
- Guardamos la posicion del ultimo parentesis de apertura (este solo es un indicador de que parte se esta evaluando de la expresión).
- Si se encuentra el parentesis de cierre.
- Si lo que esta despues de '(' es un numero.
- Mide la longitud del numero, conviértelo a entero y cambia la posicion del parentesis de apertura a donde termina el primer numero.
- Se sabe que despues del primer numero viene el operador, por tanto identificar cual es el operador (+, -, / o *). cambia la posicion del parentesis de apertura a la posicion del operador.
- Mide la longitud del segundo numero, conviértelo a entero y cambia la posicion del parentesis de apertura a donde termina el segundo numero.
- Obtenidos los dos numeros de forma entera y sabiendo cual es el operador se realiza la operación y se obtiene el resultado, dicho resultado, será el nuevo numero1, ya que, sabemos que este resultado tendra que operarse con algun otro numero más adelante.
- La posicion del parentesis de apertura se cambia a la del iterador i, sabiendo que todo lo anterior ya fue evaluado.
- El caso "else" representa el instante en que se hizo la evaluación de los parentesis mas internos y se obtuvo un resultado, este resultado se guarda como el primer numero.
- Se sabe que despues del primer numero viene el operador, por tanto identificar cual es el operador (+, -, / o *). cambia la posicion del parentesis de apertura a la posicion del operador.
- Mide la longitud del segundo numero, conviértelo a entero y cambia la posicion del parentesis de apertura a donde termina el segundo numero.
- Obtenidos los dos numeros de forma entera y sabiendo cual es el operador se realiza la operación y se obtiene el resultado, dicho resultado, será el nuevo numero1, ya que, sabemos que este resultado tendra que operarse con algun otro numero más adelante.
- La posicion del parentesis de apertura se cambia a la del iterador i, sabiendo que todo lo anterior ya fue evaluado.
- Se imprime el resultado.

```

int main (){
    char numero[255];

    introducirExpresion(numero);
    int posicionParentesisA, resultado, i;

    for(i=0; i<strlen(numero); i++){
        if (numero[i] == '('){
            posicionParentesisA = i;
        }else if(numero[i] == ')'){
            int numero1, operador, numero2, digitos;

            if (identificarOperador(numero[posicionParentesisA+1]) == 0){
                digitos = cuentaDigitos(numero, posicionParentesisA+1);
                numero1 = convertirAEntero(digitos, posicionParentesisA+1, numero);
                posicionParentesisA += digitos;
                digitos = cuentaDigitos(numero, posicionParentesisA+1);
                operador = identificarOperador(numero[posicionParentesisA+1]);
                posicionParentesisA++;
                digitos = cuentaDigitos(numero, posicionParentesisA+1);
                numero2 = convertirAEntero(digitos, posicionParentesisA+1, numero);
                posicionParentesisA += digitos;
                resultado = ejecutarOperacion(numero1, numero2, operador);
                numero1 = resultado;
                posicionParentesisA = i;
            }else{
                digitos = cuentaDigitos(numero, posicionParentesisA+1);
                operador = identificarOperador(numero[posicionParentesisA+1]);
                posicionParentesisA++;
                digitos = cuentaDigitos(numero, posicionParentesisA+1);
                numero2 = convertirAEntero(digitos, posicionParentesisA+1, numero);
                posicionParentesisA += digitos;
                resultado = ejecutarOperacion(numero1, numero2, operador);
                numero1 = resultado;
                posicionParentesisA = i;
            }
        }
    }
    printf("%d", resultado);
    return 0;
}

```

Impresiones de pantalla Linux

```
jona@jona-VirtualBox:~/Documentos$ gcc figura.c -o figura
jona@jona-VirtualBox:~/Documentos$ ./figura
Ingresa el numero de la diagonal
4
4

  *      *
 **    ** **
*****
 **    ** **
  *      *
```

```
jona@jona-VirtualBox:~/Documentos$ gcc Fibonacci.c -o Fibonacci
jona@jona-VirtualBox:~/Documentos$ ./Fibonacci
0
1
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
jona@jona-VirtualBox:~/Documentos$
```

```
jona@jona-VirtualBox:~/Documentos$ gcc parentesis.cpp -o parentesis
jona@jona-VirtualBox:~/Documentos$ ./parentesis
-----Parentesis Balanceados-----
EQUIPO 7

Ingrese la formula: 3{4(2-7)-11-2[3(-5-1)]} = 3{4 (-5)-11-2[3 (-6)]}

La formula esta balanceada

-----Parentesis Balanceados-----
EQUIPO 7

Ingrese la formula: ^C
jona@jona-VirtualBox:~/Documentos$
```

```
jona@jona-VirtualBox:~/Documentos$ gcc EvaluarOperacionAritmetica.c -o EvaluarOperacionAritmetica
jona@jona-VirtualBox:~/Documentos$ ./EvaluarOperacionAritmetica
Introduce la expresion: (((5-2)+4)*10)
70jona@jona-VirtualBox:~/Documentos$
```

Impresiones de pantalla Windows

```
C:\Users\jona-\Downloads>gcc figura.c
```

```
C:\Users\jona-\Downloads>a
Ingresa el numero de la diagonal
4
4
```

```
  *      *
 **    **
*****
 **    **
  *      *
```

```
C:\Users\jona-\Downloads>gcc Fibonacci.c
```

```
C:\Users\jona-\Downloads>a
```

```
0
1
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
```

```
-----Parentesis Balanceados-----  
EQUIPO 7  
Ingrese la formula: 2 {[3+2 (4-3)-2 (6-8)-5]} = 2 {[3+2 1-2 (-2)-5]}  
La formula esta balanceada  
Presione una tecla para continuar . . .
```

```
C:\Users\jona-\Documents\Jona\ESCOM\semestre 4\Sistemas Operativos>gcc EvaluarOperacionAritmetica.c  
C:\Users\jona-\Documents\Jona\ESCOM\semestre 4\Sistemas Operativos>a  
Introduce la expresion: (((5-2)+4)*10)  
70
```

Conclusión

En nuestra opinión es importantes saber y conocer los acontecimientos históricos acerca del desarrollo de los sistemas operativos junto con el del hardware y su representativa evolución en las últimas décadas, ya que así podemos comprender gran parte del funcionamiento de nuestros equipos. Además resaltar la importancia de los sistemas operativos y como este software base facilita la interacción entre computadora y el usuario.

Concluimos que el sistema operativo es un conjunto de programas escritos con el fin de administrar los recursos del hardware en el sistema. Si imagináramos un mundo donde el desarrollo de los sistemas operativos no hubiera emergido, podríamos imaginar un mundo diferente sin computadoras tan fáciles de manejar o incluso teléfonos inteligentes. En base a los acontecimientos ocurridos, podemos determinar que los sistemas operativos irán con el paso del tiempo evolucionando, junto con el hardware en las computadoras para seguirnos facilitando tareas para los usuarios.