USM Numérica
Licencia y configuración del laboratorio

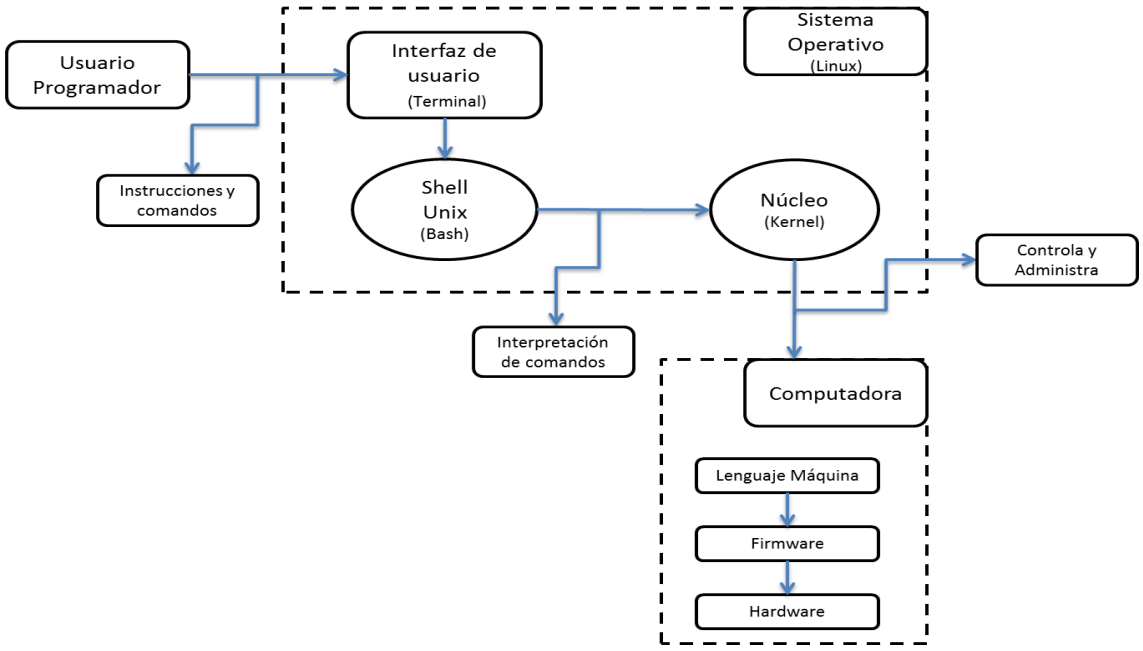
IPython Notebook v4.0 para Python 3.0

Sebastian Flores, Christopher Cooper, Alberto Rubio, Pablo Bunout.

Introducción a *BASH*

Antes de comenzar debemos saber que Bash es un programa informático usado como intérprete de comandos o instrucciones dadas por un usuario, las cuales son escritas en alguna interfaz gráfica o comúnmente una terminal. Esas instrucciones son interpretadas por Bash para luego enviar dichas órdenes al Núcleo o Kernel del sistema operativo.

Cada sistema operativo se encuentra conformado por un Núcleo particular, que se encarga de interactuar con la computadora siendo una especie de cerebro capaz de organizar, administrar y distribuir los recursos físicos de esta, tales como memoria, procesador, forma de almacenamiento, entre otros.



Bash (Bourne-Again-Shell) es un lenguaje de programación basado en Bourne-Shell, el cual fue creado para sistemas Unix en la década de los 70, siendo el sustituto natural y de acceso libre de este a partir del año 1987 siendo compatible con la mayoría de sistemas Unix, GNU/Linux y en algunos casos con Microsoft-Windows y Apple.

Objetivos

1. Operaciones básicas para crear, abrir y cambiarse de directorio
2. Operaciones para crear un archivo, copiar y cambiarlo de directorio
3. Visualizador gráfico de directorios y archivos
4. Visualizador de datos y editor de un archivo de texto
5. Ejercicio de práctica

1. Operaciones para crear, abrir y cambiar de directorio

Este es el tipo de operaciones más básicas que un usuario ejecuta en un sistema operativo, los siguientes comandos nos permiten ubicarnos en alguna carpeta para tener acceso a algún archivo o material en específico, crear carpetas o directorios para almacenar información deseada entre otros.

La acción más simple para comenzar será ingresar a un directorio o carpeta deseada usando el comando *cd* como sigue:

```
cd <directorio>
```

Una extensión de este recurso es la posibilidad de colocar una secuencia de directorios para llegar a la ubicación deseada, separando los nombres por un slash del siguiente modo.

```
cd <directorio_1>/<subdirectorio_2>/<subdirectorio_3>
```

Podemos visualizar en la terminal el contenido de este directorio con el comando *ls* y luego crear un nuevo sub-directorio (o carpeta) dentro del directorio en el cual nos ubicamos con *mkdir*:

```
mkdir <subdirectorio>
```

Mencionamos además una opción con el comando anterior, que nos permite crear varios sub-directorios a la vez escribiendo sus nombres respectivos uno al lado del otro, separados por un espacio.

```
mkdir <subdirectorio_1> <subdirectorio_2> ... <subdirectorio_N>
```

Si queremos regresar a una ubicación anterior basta los comandos *cd ..* o *cd -* y si queremos volver al directorio original desde donde se abrió la terminal usamos *cd ~*.

Es posible borrar un directorio con su contenido al interior escribiendo el siguiente comando:

```
rm -r <directorio>
```

Finalmente un comando que nos permite visualizar rápidamente nuestra ubicación actual y las precedentes es *pwd*.

2. Operaciones para crear, copiar y eliminar archivos

Un paso siguiente a lo visto en el punto anterior es la creación de algún tipo de archivo, realizando operaciones básicas como copiarlo de un directorio a otro, cambiarlo de ubicación o borrarlo.

Para crear un archivo debemos ingresar al directorio en el cual deseamos guardarlo con el comando *cd* y luego de esto podemos crear el archivo con el argumento *>* de la siguiente manera.

```
> <archivo.tipo>
```

Por defecto el archivo se crea en el directorio actual de nuestra ubicación, recordar que con pwd podemos visualizar la cadena de directorios y subdirectorios hasta la ubicación actual. Debemos hacer referencia al comando *echo*, este consiste en una función interna del intérprete de comandos que nos permite realizar más de una acción al combinarlo de distintas maneras con otros comandos o variables. Uno de los usos más comunes es para la impresión de algún texto en la terminal.

```
echo <texto a imprimir>
```

También nos permite imprimir un texto en un archivo específico agregando *echo <texto a imprimir> < <archivo sobre el que se imprime>*, entre muchas otras opciones que la función *echo* nos permite realizar y que es posible profundizar con la práctica, pero estas las postergaremos para la siguiente sección. Continuamos con el comando *mv*, que refiere a "move", el cual sirve para mover algún archivo ya creado a un nuevo directorio.

```
mv <archivo.tipo> <directorio>
```

También sirve para mover un directorio dentro de otro (mover *directorio_1* al *direcotorio_2*), para que el comando se ejecute correctamente ambos directorios deben estar en una misma ubicación.

```
mv <directorio_1> <directorio_2>
```

Una operación similar a la anterior es copiar un archivo y llevarlo a un directorio particular, con la diferencia que una vez realizada la acción se tendrán 2 copias del mismo archivo, una en el directorio original y la segunda en el nuevo directorio.

```
cp <archivo.tipo> <directorio>
```

Del mismo modo que para un directorio, si queremos borrar un archivo creado podemos hacerlo con el comando *rm -r*.

```
rm -r <archivo.tipo>
```

Y si queremos borrar una serie de archivos lo hacemos escribiendo consecutivamente.

```
rm -r <archivo_1.tipo> <archivo_2.tipo> ... <archivo_N.tipo>
```

3. Visualizador de estructura de directorios y archivos

El comando *tree* es una forma útil y rápida de visualizar gráficamente la estructura de directorios y archivo pudiendo ver claramente la relación entre estos. Solo debemos escribir el comando para que automáticamente aparezca esta información en pantalla (dentro de la terminal) apareciendo en orden alfabético, por defecto debe ejecutarse ubicándose en el directorio deseado visualizando la estructura dentro de este.

En caso de que este no se encuentre instalado en nuestro sistema operativo, a modo de ejercicio, primero debemos escribir los siguientes comandos.

```
sudo apt-get install <tree>
```

4. Visualizar, editar y concatenar un archivo de texto

Para visualizar el contenido de un texto previamente creado, pudiendo hacerlo con el comando visto anteriormente, *echo > arhcivo.tipo*, utilizamos el comando *cat*.

```
cat <archivo.tipo>
```

Luego si queremos visualizar varios archivos en la terminal, lo hacemos agregando uno al lado del otro después del comando *cat*.

```
cat <archivo_1.tipo> <archivo_2.tipo> ... <archivo_N.tipo>
```

Existen muchos argumentos que nos permiten visualizar de distinta forma el contenido de un archivo en la terminal, por ejemplo enumerar las filas de algún texto, *cat -n*, otra opción sería que solo se enumerara las filas que tienen algún contenido, *cat -b*.

En caso de que queramos enumerar solo las filas con texto, pero este tiene demasiadas filas en blanco y buscamos reducirlas a una sola de modo de ahorrar espacio en la terminal, podemos hacerlo agregando el argumento *-s* como sigue.

```
cat -sb <archivo.tipo>
```

Editar o imprimir un texto en un archivo es posible hacerlo usando la función *echo* como sigue.

```
echo <texto a imprimir> ./archivo.txt
```

Similar a sudo, less es un programa usado como visualizador de archivos de texto que funciona como un comando interpretado desde la terminal. Este permite visualizar completamente el archivo de texto usando por defecto las flechas del teclado para avanzar o retroceder en el visualizador.

Una de las ventajas de un programa como less, es que puede añadirse comandos para ejecutar acciones de forma rápida en modo de comandos que resulta por defecto al ejecutar less, a continuación presentamos algunos comandos básicos.

```
G: permite visualizar el final del texto

g: permite visualizar el inicio del texto

h: nos proporciona ayuda respecto a comandos posibles

q: permite salir de la aplicación dentro del visualizador less
```

Para modificar el texto una de las formas es cargar algún editor de texto como por ejemplo el Visual.

```
v: ejecutar el editor de texto
```

5. Ejercicio de práctica

Para redondear lo visto en este tutorial se dejara como ejercicio las siguientes instrucciones:

- Crear una carpeta o directorio principal
- En ella se debe copiar 2 archivos de textos provenientes de cualquier dirección
- Crear un archivo de texto el cual tenga por nombre "Texto Principal" y se imprima "concatenación de textos"
- Crear una segunda carpeta dentro de la principal
- Concatenar los 2 archivos copiados con el archivo creado
- Mover el archivo "Texto Principal" a la nueva carpeta
- Eliminar las copias de los archivos concatenados
- Visualizar con Tree la estructura y relación de archivos y directorios creados