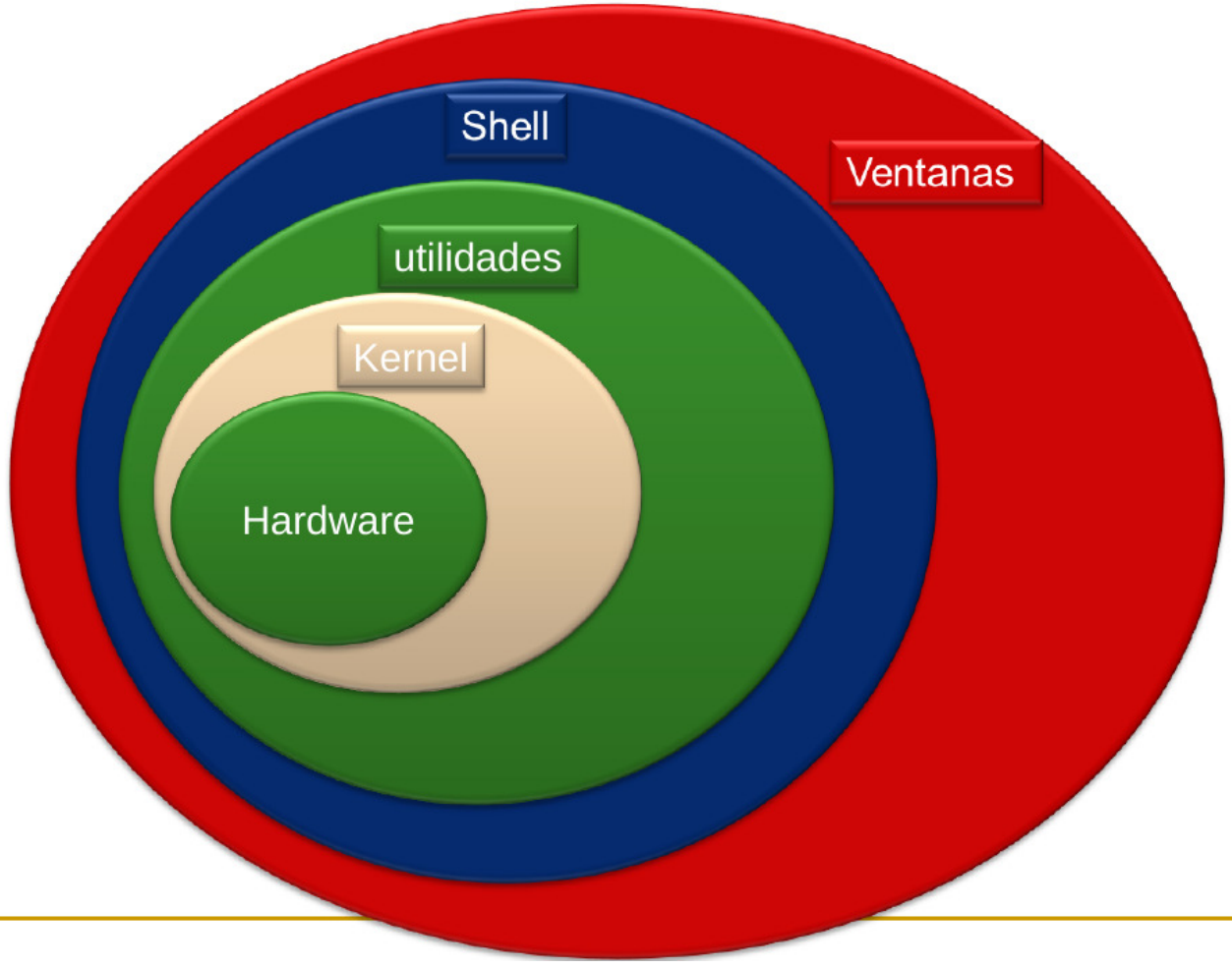
Ya esta funcionando la página web de la materia:
`https://pulidom.github.io/programacion`

Recuerden que los Lunes vamos a tener clase cuando sea necesario recuperar teoría o cuando tengamos prácticos retrazados.

Sistema Operativo



El sistema operativo propiamente es el kernel (nucleo), el resto incluyendo el administrador de archivos, bash, administrador de pantalla son aplicaciones.

Distribuciones del linux

- Debian
- Slackware
- Opensuse (HPC/Servidores).
- Ubuntu
- Lubuntu (Light-ubuntu).
- Centos, Fedora, Arch Linux, Mandriva, etc

Organización de los directorios y usuarios del linux

- user: Puede configurar sus propias aplicaciones y utilizar el sistema sin realizar cambios.
- root (superuser): Es el unico que tiene permiso de cambiar variables del sistema operativo.

Estructura de directorios principales

- /root
- /home
- /usr
- /media
- /etc
- /proc

/home/[nombre-usuario] Es nuestro directorio de cabecera y alli tendremos nuestra estructura interna.

Permisos de los archivos

Permisos de los directorios o archivos: Lectura(r), Escritura(w), Ejecución(x)

El único lugar que tendremos permiso de escritura es el /home/[nombre-usuario]
En el resto de los directorios generalmente se tiene permiso de lectura pero no de escritura.

Hay archivos que pueden ejecutarse (con instrucciones que entienda la máquina). Todos los directorios son ejecutables.

Terminal: bash/c-shell

Es un lenguaje de programación que sirve para manejar a la computadora a través de cómandos básicos. Generalmente, se utiliza para configurar los sistemas, para manejar los dispositivos, manejo de usuarios, etc.

Lenguajes de programación: Interpretés y Compiladores.

Existen dos tipos de forma de dar las instrucciones en una terminal bash:

1. Interactiva: Ejecutan cada instrucción.
2. Script: Es un programa (conjunto de instrucciones) que se guarda en un archivo. El archivo/file debe tener permiso de “ejecución”.

ls: list

El comando ’ls’ produce una lista de los archivos del directorio donde nos encontramos.

```
$ ls [directorio]
```

nos lista los archivos dentro del directorio especificado.

```
$ ls [file]
```

nos lista el file especificado (si es que se encuentra alli o da un error si el file no existe).

Modificatorios/Opciones del comando:

```
$ ls -t
```

Lista los archivos ordenados por el último modificado primero.

Generalmente las opciones o modificatorios van con un ’-’

Atributos de un archivo

```
$ ls -l
```

Realiza la lista de archivos con los atributos de cada archivo.

Atributos de un archivo:

- Permisos (lectura, escritura, ejecucion).
- Usuario y grupo (de usuarios) del archivo
- Tamaño de los arhivos en bytes.
- Última fecha de modificación del file

El asterisco (comodin del poker)

```
$ ls a*
```

Lista todos los archivos que comienzan con ’a’

```
$ ls /usr/*.txt
```

Lista todos los archivos con extensión txt que se encuentran en el directorio usr

```
$ ls /usr/a*.txt
```

Igual que el anterior pero solo los que empiezan con a

cd: Change Directory

cd cambia el directorio.

Si queremos ir a un directorio específico:

```
$ cd /usr/local
```

Si queremos volver en un directorio de /usr/local a /usr:

```
$ cd ../
```

(los dos puntos vuelven para atras en la estructura de directorio).

Si queremos ir a nuestro home directory /home/[usuario] usar la tilde:

```
$ cd
```

o

```
$ cd ~
```

Si queremos ir a algun directorio de nuestro home,

```
$ cd ~/programacion/guia1
```

(la tilde es una abreviatura de /home/usuario)

Mover, copiar, borrar archivos

Para mover un archivo o directorio: mv (move)

```
$ mv [file] [directory]
```

```
$ mv [directory] [directory]
```

Para copiar un archivo o directorio: cp (copy)

```
$ cp [file] [directory]
```

Para borrar archivos o directorio: rm (remove) NO se recomienda

Elimina totalmente los archivos o directorios (no se pueden recuperar).

Alternativa es: mover al trash

```
$ rm -i [file]
```

La opción -i hace interactivo (pregunta si realmente lo quieren borrar).

Varios

- man ls: man=manual, nos da información sobre el comando, la instrucción.
- mkdir Crea un directorio. MaKe DIRectory
- rmdir Crea un directorio. MaKe DIRectory
- pwd: muestra el nombre del directorio de trabajo (Print Working Directory)
- touch: Si el archivo no existe, lo crea. Si ya existía le cambia el tiempo al actual.
- echo: Imprime/Muestra en pantalla. echo 'Hola' echo Hola echo $HOLA
  Cualquier palabra con un $ se esta refiriendo a una variable del sistema $var (variable var). Entonces imprime el contenido de la variable 'HOLA' (si no esta definida no imprime nada).

Comando chmod

chmod: Change file mode. Cambia los permisos de un archivo. Obviamente solo nos permite cambiar los permisos de los archivos que somos propietarios.

chown: Change owner nos permite cambiar al propietario de un archivo

```
$ chown usuario:users programacion
```

[Hace que programación pertenezca al 'usuario' y al grupo 'users'] (solo para archivos dentro de nuestro directorio)

Comando find

find Encontrar, es decir busca/encuentra archivos o directorios.

```
$ find /home/[usuario] -name "*.txt"
```

En este caso busca todos los archivos que tengan como extensión txt en nuestro home directory.

Comando grep

grep Busca en un archivo o grupo de archivos, una palabra o patrón.
Sintaxis: `$ grep -[opciones] [patron] [archivo]`

Supongamos queremos encontrar en un conjunto de archivos donde tenemos la información del experimento “Maxwell”.
Entonces lo que hacemos es buscar el patron: ’Maxwell’ en los directorios exp*.txt.

```
$ grep ’Maxwell’ exp*.txt
```

Este comando imprimirá como salida todas las líneas en los archivos exp*.txt que tengan la palabra Maxwell (El linux distingue entre mayúsculas y minúsculas)
Si se quiere buscar palabras con mayúsculas o minúsculas, maxwell, MAXWELL o Maxwel, agregar la opción -i.

Redireccionamiento de la salida de un comando

Existen comandos para redireccionar la salida de un comando.

'>' Manda la salida de un comando a un archivo:

Ejemplo:

```
$ ls > listado.txt
```

Guarda la salida del comando en un archivo llamado listado.txt. Si no esta lo genera al archivo. De lo contrario lo sobreescribe.

Ejemplo:

```
$  ls *.txt | grep 'Ejercicio'
```

Busca en todos los archivos .txt la palabra Ejercicio. Es decir lo que hace el comando | es pasarle la lista de archivos encontrados al 'grep'.

Script

Para generar un script abrimos/generamos un archivo, prueba.sh (la extensión .sh nos indica por convención que esta en bash/shell, pero se puede usar cualquier cosa)

Luego conviene poner en la primera línea del archivo:

```
#!/bin/bash
```

Esta instrucción dice que lo que viene esta en lenguaje de programación 'bash'.

Script para que la computadora salude y se presente: saludo.sh

```
#!/bin/bash
echo 'Hola. Soy Alexa'
```

Por otro lado se deben asegurar de que el archivo tenga permiso de ejecución.

```
$ chmod a+x saluda.sh
```

Luego para ejecutarlo solo tienen que hacer:

`$ ./saluda.sh` (si estan en el mismo directorio)

Si estan en cualquier lugar:

```
$ ~/programacion/saluda.sh
```

(Se pone todo el path a camino)

Scripts con variables de entrada

Modificamos el programa anterior para que interactue con el usuario.

```
#!/bin/bash
echo 'Hola '$1'. Yo soy Alexa'
```

Ejecutando como: ./saluda.sh [tuNombre]

Es decir, que $1 es una variable de entrada en el script.

Cálculos con el bash

Los cálculos en bash NO son recomendados. Vamos a utilizar python que es mucho mas potente. Bash tiene una forma bastante primitiva de cálculos:

```
$ myvar = 2
$ expr $myvar + 1
$ expr $myvar / 3
$ expr $myvar \* 3
```