DESARROLLO WEB FULL STACK

Terminal

Entendiendo el intérprete de comandos

DWFS COR

¿Que es la terminal?

Terminal o consola es un **intérprete** de comandos en **modo texto**, es decir, sólo acepta comandos ingresados por teclado. Nos permite ejecutar tareas directamente al núcleo del sistema operativo.

La terminal, con los permisos adecuados, tiene accesos a cualquier configuración del sistema operativo y muchas herramientas de desarrollo solo existen en modo terminal.

También permite crear scripts para automatizar grupos de comandos para simplificar aún más los flujos de trabajo.

Modo Texto:
Es un tipo de interfaz que nos permite interactuar con nuestro sistema mediante el teclado y una consola compuesta solo por caracteres.

Interprete de comandos:
Es un software que traduce comandos u órdenes del usuario directamente a las herramientas y núcleo del sistema operativo. Por lo cual es una herramienta en sí misma muy potente pero muy peligrosa.

Terminales existentes

Unix	**Windows**
Korn Shell (ksh)	command.com
C Shell (csh)	cmd.exe
Bourne Shell (sh)	MSH Windows PowerShell (Monad)
Bourne Again Shell (bash)	
Z Shell (Zsh)	

Comandos

Los comandos Bash son conjunto de parámetros utilizados para la administración y configuración del sistema, así como un conjunto de combinaciones especiales de teclas para realizar tareas específicas en entornos Linux/Unix mediante un intérprete de comandos Bash.

Lista de comandos:
https://courses.cs.washington.edu/courses/cse391/17sp/bash.html

Referencia:
https://es.wikipedia.org/wiki/Comandos_Bash

Algunos comandos

Listado de comandos básicos.

```
$ # cd : comando para cambiar de directorio
$ # con tab podemos autocompletar comandos y rutas
$ cd

$ # ls : Nos lista todas las carpetas y archivos
$ ls

$ # pwd : nos muestra en que carpeta nos encontramos
actualmente.
$ pwd

$ # mkdir : crea una nueva carpeta vacía
$ mkdir <nombre_carpeta>

$ # rm : borra un archivo o carpeta -rf (recursivo)
$ rm <nombre_carpeta> | rm -r <nombre_carpeta>

$ # touch : crea un archivo vacío.
$ touch <nombre_archivo>
```

Algunos comandos

Listado de comandos un poco más avanzados

```
$ # grep : Busca en archivos de un directorio por string o expresión
$ grep

$ # find : Busca archivos en un directorio según string o expresión
$ find

$ # locate : Busca según string o expresión archivos en todo el sistema.
$ locate

$ # echo : Imprime una cadena de texto por terminal.
$ echo

$ # nano : Lanza el editor de texto básico.
$ nano <nombre_archivo>
```

Algunos comandos

Comandos de control de flujo

```
$ # Todos sabemos lo que hace :)
$ if
$ else
$ while

$ # Copia la salida de texto de un comando a un archivo
$ # > (sobreescribe), >> (adiciona)
$ >
$ >>

$ # pipe (Envía la salida de un comando hacia el siguiente
comando)
$ |
```

Ejemplos

Algunos ejemplos

```
$ echo “Salida por terminal :)”

$ cd Documentos

$ echo “Algo que quiero copiar al archivo” > archivo.txt

$ touch memes.txt

$ nano memes.txt

$ grep “thanos” memes.txt

$ ls | grep “memes”

$ ls | find *.txt

$ grep “thanos” memes.txt | xargs echo > thanos.txt

$ find memes.txt | xargs grep “thanos” > memix.txt
```

Ejercicio práctico:
Mediante la terminal, crear una carpeta nueva “practica_bash” ingresar a la misma y crear un archivo busqueda.sh y abrirlo con el editor de texto “nano”.

Shell scripting

Un **script de shell** es un programa de computadora diseñado para ser ejecutado por el shell de Unix, un intérprete de línea de comandos.

Nos permite automatizar tareas y no tener que escribir los comandos de bash nuevamente todas las veces necesitemos realizar una determinada tarea.

Muy buen tutorial:
https://www.howtoforge.com/tutorial/linux-shell-scripting-lessons/

En un file busqueda.sh

```
#!/bin/bash


if [ -e memes.txt ]
then
    echo "el archivo existe!"
else
    echo "el archivo no existe"
fi
```

Ejecutando nuestro shell script

Lo ejecutamos

```
$ # Damos permisos de ejecución al archivo
$ chmod 777 busqueda.sh

$ # Lo ejecutamos
$ ./busqueda.sh
```

En un file busqueda.sh

```
#!/bin/bash


read -p "Ingrese el nombre del archivo: " filename
if [ -e $filename ]
then
    echo "el archivo existe!"
else
    echo "el archivo no existe"
fi
```

En un file busqueda.sh

```
#!/bin/bash


if [ -e $1]
then
    echo "el archivo existe!"
else
    echo "el archivo no existe"
fi
```

Ejercicio final:
Crear un shell script que cree un archivo de texto con el nombre que se le pase por primer parámetro y el contenido que se le pase por segundo parámetro.