

Instituto Politécnico Nacional

Escuela Superior de Cómputo

* Alumnos:

1. Arcos Hermida Aldo Alejandro (N° lista 5)
2. Chávez Becerra Bianca Cristina (N° lista 9)
3. Islas Osorio Enrique (N° lista 20)
4. Juárez Cabrera Jessica (N° lista 21)
5. Palmerin García Diego (N° lista 28)

* Grupo: 4CM1
* Materia: Sistemas operativos
* Profesor: Araujo Díaz David

Practica 1

Introducción al shell: comandos básicos y scripts

**Practica 1: Introducción al shell: comandos básicos y scripts**

**Objetivo**

Aprender el uso de los comandos básicos de interpreté de comandos de LINUX/UNIX, así como la ejecución de scripts.

**Descripción**

Los usuarios pueden interactuar con el sistema operativo por medio de un intérprete de comandos (Shell) o una interfaz gráfica de usuario (GUI). El intérprete de comandos es un programa del sistema que proporciona una interfaz de línea de comandos, mediante la cual el usuario indica al sistema operativo lo que se quiere hacer.

Existen diferentes intérpretes de comandos: Bourne shell (sh), C shell (csh), Bourne again shell (bash), korn shell (ksh), tenex c shell (Tcsh) y Zero shell (zsh). En la mayoría de estos el indicador del sistema consiste en el nombre del equipo, seguido por dos puntos (:), el directorio actual seguido por un carácter que indica el tipo de usuario conectado. Los usuarios pueden ser de tipo administrador (root) representado por un # o usuario normal denotado por $.

* **Consola**

La interfaz de línea de comandos o consola (en inglés, *command-line interface*, CLI) es un tipo de interfaz de usuario de computadora que permite a los usuarios dar instrucciones a algún programa informático o al sistema operativo por medio de una línea de texto simple.

En su forma más simple, una CLI consiste en un espacio donde se pueden escribir órdenes las cuales se las manda a un módulo interpretador de órdenes (Shell) que analiza la secuencia de caracteres recibida y, si la sintaxis de la orden es correcta, ejecuta la orden dentro del contexto del programa o del sistema operativo donde se encuentra.

* **Shell**

El shell es un programa que te permite ejecutar otros programas. También se dice que es un intérprete de comandos, que al fin de cuentas esos comandos son programas que realizan alguna acción en el equipo.

* **Bash**

Es una popular interfaz de usuario de línea de comandos, específicamente un shell de Unix; así como un lenguaje de scripting. Bash fue originalmente escrito por Brian Fox para el sistema operativo GNU.

Bash es un intérprete de órdenes que generalmente se ejecuta en una ventana de texto donde el usuario escribe órdenes en modo texto. Bash también puede leer y ejecutar órdenes desde un archivo, llamado guión o 'script'. Al igual que todos los intérpretes de Unix, es compatible con variables y estructuras de control para pruebas de condición e iteración.

* **Scripting**

Los #! es la sintaxis que se utiliza en los scripts de shell para indicar al intérprete que ejecute el script en los sistemas operativos Unix / Linux. Además especifica que el script debe ejecutarse usando bash como intérprete, y el intérprete bash se puede encontrar en el directorio /bin.

|  |
| --- |
| *#!/bin/bash* |

Para comenzar a ejecutar un script en Unix/Linux se requiere cambiara el permiso que inicialmente tiene nuestro archivo. Para eso ejecutamos el siguiente comando:

|  |
| --- |
| $ chmod 777 hello-world.sh |

Ciertamente, depende de cuál sea la funcionalidad del script se debe de otorgar los permisos al usuario. Pero para fines prácticos se coloca 777 para indicarle que tiene permiso de leer, escribir y ejecutar el usuario, los grupos y todos.

Ahora si ejecutamos el archivo de la siguiente manera:

|  |
| --- |
| ./hello-world.sh |

* **Echo**

|  |
| --- |
| **#!/bin/bash** **echo** "Hello world" |

**Salida**



* **IF statement**

Básicamente la estructura de control If funciona como el comando test, por lo tanto cualquier operación lógica que queramos realizar tendrá que hacer utilizando los operadores que con este viene.

|  |
| --- |
| **#!/bin/bash** **# Condiciones if** **if [ $1 -gt 500 ]** **then**  **echo Es un número grande** **elif [ $1 -gt 100 ]** **then**  **echo Es un número medio** **else**  **echo Es un número chico** **fi** |

El signo $1 hace referencia al argumento con el que ejecutamos nuestro script, es decir:

|  |
| --- |
| ./condiciones.sh <numero> |

Entonces dependiendo de qué número hayamos ingresado $1 tomará ese valor.

Cabe aclarar que lo que se pone dentro de los corchetes es otra forma de escribir el comando de test. Si hubiéramos escrito lo mismo, pero con una sintaxis diferente también se ejecutará sin problemas.

**Salida**







|  |
| --- |
| **#!/bin/bash** **# Condiciones if** **if** [[ $1 -gt 500 ]] then  echo Es un numero grande elif test $1 -gt 100 then  echo Es un numero medio else  echo Es un numero chico fi |

Algunos ejemplos de operadores lógicos disponibles son:

|  |  |
| --- | --- |
| Operator | Description |
| ! EXPRESSION | The EXPRESSION is false. |
| -n STRING | The length of STRING is greater than zero. |
| -z STRING | The lengh of STRING is zero (ie it is empty). |
| STRING1 = STRING2 | STRING1 is equal to STRING2 |
| STRING1 != STRING2 | STRING1 is not equal to STRING2 |
| INTEGER1 -eq INTEGER2 | INTEGER1 is numerically equal to INTEGER2 |
| INTEGER1 -gt INTEGER2 | INTEGER1 is numerically greater than INTEGER2 |
| INTEGER1 -lt INTEGER2 | INTEGER1 is numerically less than INTEGER2 |
| -d FILE | FILE exists and is a directory. |
| -e FILE | FILE exists. |
| -r FILE | FILE exists and the read permission is granted. |
| -s FILE | FILE exists and it's size is greater than zero (ie. it is not empty). |
| -w FILE | FILE exists and the write permission is granted. |
| -x FILE | FILE exists and the execute permission is granted. |

Para realizar una operación lógica como lo sería and (&&) o or (||). Se ejecuta de la siguiente manera:

|  |
| --- |
| **if** [ $1 -gt 500 ] || [ $1 -lt 1000 ] |

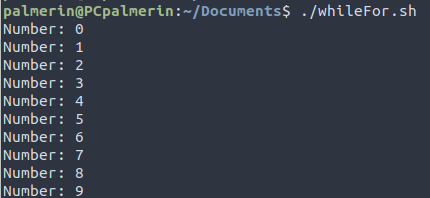


* **While y For**

Al igual que los if statement while se requiere de una condición de paro para eso se utiliza el comando de test.

|  |
| --- |
| **#!/bin/bash** **# Estructuras de iteración** **i=0** **while [ $i -lt 10 ]** **do**  **echo Number: $i**  **((i++))** **done** |

Salida



El uso más común para el while, es para leer ya sea archivos o data stream. Un ejemplo de esto es:

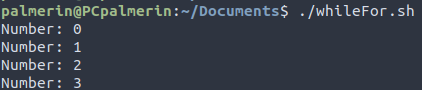
|  |
| --- |
| **#!/bin/bash** **file=/etc/passwd**  **while** read -r line; **do** echo $line done < "$file" |

Este programa imprimirá todo lo que haya en el archivo passwd ubicado en el directorio de etc, línea por línea.



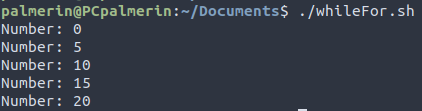
La estructura for es muy similar, por ejemplo: el siguiente programa imprime los números del 0 al 3

|  |
| --- |
| **#!/bin/bash** **for** i **in** {0..3} do  echo "Number: $i" done |



Para realizar un incremento con otro valor se hace de la siguiente manera:

|  |
| --- |
| **#!/bin/bash** **for** i **in** {0..20..5} do  echo "Number: $i" done |



* **Comandos**

Existen una gran variedad de comandos. Algunos pueden variar en su sintaxis según la distribución que se esté utilizando. En este caso se usó Ubuntu para probar los comandos más básicos, pero muy importantes, pues también son de los más utilizados.

cal (calendario) – muestra el calendario

Texto

Descripción generada automáticamente

date

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

mkdir, cd, pwd

* mkdir (make directory) – Crea un nuevo directorio
* cd (change directory) – Cambia del directorio actual a otro. Similar a seleccionar una carpeta.
* pwd (print working directory) – Muestra el directorio actual

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

rmdir (remove directory) – elimina un directorio

Texto

Descripción generada automáticamente

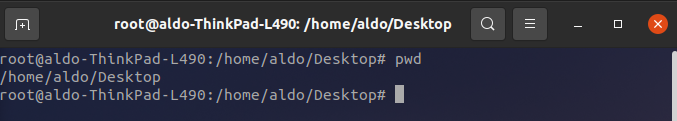
ls – muestra los contenidos del directorio

Texto

Descripción generada automáticamente con confianza media

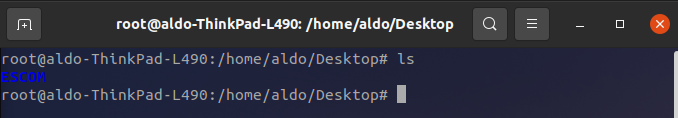
**pwd**: Muestra la ubicación del directorio actual:

|  |
| --- |
| pwd |



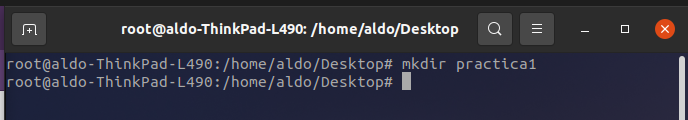
**ls**: Imprime el contenido de un directorio:

|  |
| --- |
| ls |



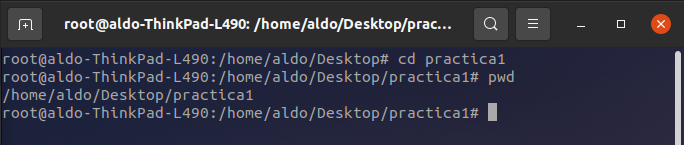
**mkdir**: Crear una carpeta, se nombra “practica1”:

|  |
| --- |
| mkdir practica1 |



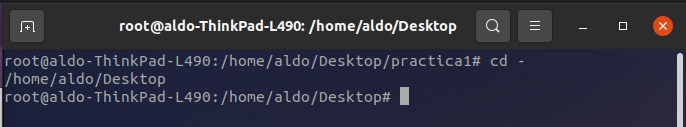
**cd**: Cambiar al directorio “practica1”:

|  |
| --- |
| cd practica1 |



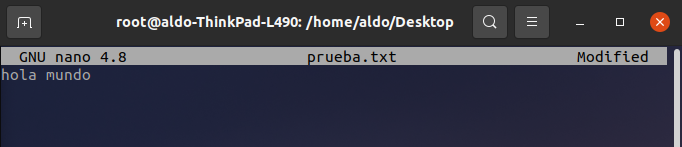
**cd**: Regresar al directorio anterior inmediato:

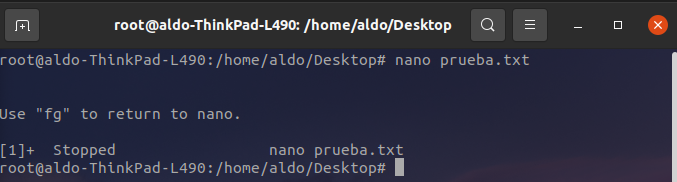
|  |
| --- |
| cd - |



**nano**: crear el fichero “prueba.txt, guardar y salir del editor nano:

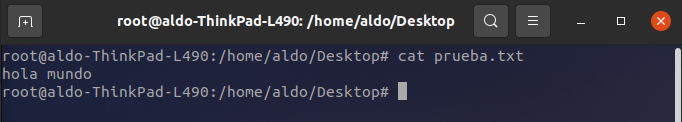
|  |
| --- |
| nano prueba.txt //ctrl + o //ctrl + z |





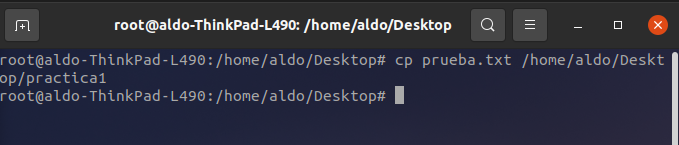
**cat**: Ver el contenido de “prueba.txt”:

|  |
| --- |
| cat prueba.txt |



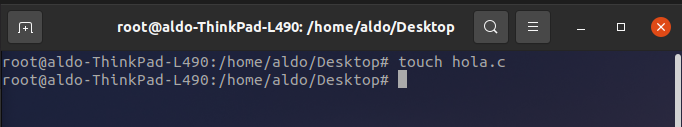
**cp**: Copia el archivo ”prueba.txt” a la carpeta “practica1”:

|  |
| --- |
| cp prueba.txt /home/aldo/Desktop/practica1 |



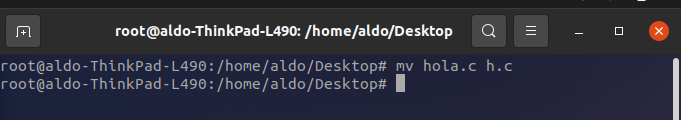
**touch**: Crear un archivo .c:

|  |
| --- |
| touch hola.c |



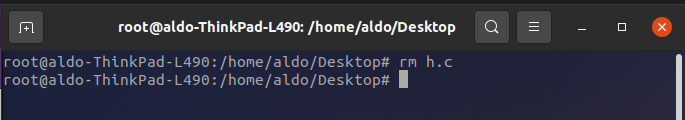
**mv**: Renombrar el archivo “hola.c” a “h.c”:

|  |
| --- |
| mv hola.c h.c |



**rm**: Borrar el archivo “h.c”:

|  |
| --- |
| rm h.c |



* **Conclusiones**

Por medio de esta práctica lo logro entender e implementar algunos de los comandos basicos y la ejecución de script. Los comandos nos ayudan a saber en donde estamos actualmente, ver la lista de los ficheros, directorio raíz del sistema, creación de directorios y ficheros, copiar o ficheros a otras carpetas, borrar ficheros, pero en Linux no se pueden borrar carpetas directamente primero se necesita borrar el contenido y después borrar el directorio o de manera más directa con banderas para borra todo dentro del directorio en caso de ser muchos archivos

Expresa sus conclusiones de la realización de la práctica. Resume los resultados. Presenta los principios, relaciones y generalizaciones de los resultados. Muestra las excepciones y la correlación entre puntos. Muestra los resultados y la interpretación de estos. Discute las implicaciones teóricas y las aplicaciones prácticas. Muestra las conclusiones tan claramente como sea posible. Resume la evidencia de cada conclusión.

**Referencias**

* GeeksforGeeks. (2021, 18 febrero). *Essential Linux/Unix Commands*. Recuperado 22 de febrero de 2022, de <https://www.geeksforgeeks.org/essential-linuxunix-commands/>
* Jurado, C. L. (2021, 18 febrero). *¿Qué es la «shell» de Linux?* CCM. Recuperado 22 de febrero de 2022, de <https://es.ccm.net/contents/316-linux-shell>
* Silberschatz, A., Baer, G. & Gagne, G. (2018) Operating Systems Concepts. Wiley
* Valois, M. A. (2021, 27 julio). *Comandos básicos Linux: es hora de dar tus primeros pasos*. Blog HostGator México. Recuperado 22 de febrero de 2022, de <https://www.hostgator.mx/blog/comandos-basicos-linux/>