# Sistemas Operativos

*TRABAJO PRÁCTICO LABORATORIO Nº 1: Comandos Unix y Shell scripts*

## Comandos Básicos en Linux

1) Explica en no más de un renglón cada uno de los siguientes comandos:
    a. Relacionados con archivos: ls, cat, more, less, cp, rm, mkdir, cd, pwd, file, df, grep.
    b. Relacionados con la documentación del sistema: whatis, info, man.
    c. passwd, mount, time, date, vi, gcc, exit, reboot, halt, poweroff
    d. Relacionados con procesos y memoria: ps, kill, top
    e. Relacionados con redes: ifconfig, ping, ip, ssh, telnet, ftp

2) Menciona al menos cinco comandos equivalentes entre Unix y Windows. Por ejemplo, ***ls ≈ dir***. Intenta preferentemente con los relacionados a redes.

3) Explica (con la ayuda del manual en línea) y prueba el comando chmod.
    a. ¿Cómo invocarías ***chmod*** para asignar permisos de lectura, escritura y ejecución al propietario, lectura y ejecución para el resto de los usuarios del sistema? Supón que estos permisos deben asignarse a todos los archivos contenidos en una carpeta de nombre *scripts* (sin entrar a la carpeta).
    b. ¿Cómo invocarías ***chmod*** para asignar permisos de lectura y escritura (sin ejecución) al propietario y solo lectura para el resto de los usuarios del sistema? Supón que estos permisos deben asignarse a todos los archivos con extensión *.txt* en el directorio actual.

## Shell Scripts

Resuelve los siguientes ejercicios en Linux.
**Observación:** Los códigos fuentes de los ejercicios de interpretación, se encuentran en nuestro repositorio en GitHub.

1) Dado el siguiente script:

```
1  #! /bin/bash
2  # ejercicio 1
3
4  clear
5  while :
6  do
7          echo -e "\033[H"
8          ps
9  done
```

   a. Describir qué hace. ¿Qué comandos usa? ¿Qué hace cada comando?
   b. Modificar el script de forma tal que ejecute cualquier comando, en vez de hacerlo sólo con ps.

2) Construir un script que reciba argumentos e imprima cada uno en líneas separadas utilizando la estructura de repetición while.

3) Rehacer el script del punto 1, utilizando un for en lugar de while.

4) Desarrollar un script que reciba un nombre de archivo como primer argumento y ordene el contenido del mismo línea a línea, alfabéticamente de forma ascendente o descendente, de acuerdo a si el segundo parámetro es '-A' o '-Z'. Verificar que el script reciba los 2 argumentos, sino mostrar la forma de uso. Ejemplo de uso:

**$ ./nombre_script.sh [archivo] [-A|-Z]**

5) Dado el siguiente shellscript:

```
1  #!/bin/bash
2  #
3
4  if [ $# -gt 0 ]; then
5          if [ $# -ne 1 -o $1 != "-h" ]; then
6                  echo "Forma de uso: $0 [-h]"
7                  exit 1
8          fi
9
10         for i in *
11         do
12                 lfn=`echo $i | gawk ' {str = tolower($0);print str }'`
13                 if [ $lfn != $i ]; then
14                         mv $i $lfn
15                 fi
16         done
17 else
18         echo "Forma de uso: $0 [-h]"
19         exit 1
20 fi
```

Mediante inspección del código, describir qué hace. Luego transcribir en Linux y probar. ¿Cómo podría mejorarse?
**NOTA:** En la línea 16 se asigna a la variable *lfn* el contenido de la variable *i* transformado a minúsculas.

6) Dado el siguiente ShellScript:

```
1  #!/bin/bash
2  # shscript
3
4          toggle="A"
5          if [ "$1" = "-p" ]; then
6                  toggle="B"
7          fi
8          while read line
9          do
10                 if [ "$toggle" = "A" ]; then
11                         echo $line
12                         toggle="B"
13                 else
14                         toggle="A"
15                 fi
16         done
```

a. Describir qué hace. ¿Cuál es la funcionalidad de la variable *toggle*?
b. Probar redireccionando en la entrada un archivo con texto.

7) El siguiente shellscript está pensado para recibir uno o dos parámetros en línea de comandos, ni más ni menos:

```
#! /bin/bash
#

if [ $# -lt 1 -o $# -gt 2 ]; then
        echo "Cantidad incorrecta de parámetros"
        exit 1
fi

if [ "$1" = "-" ]
        then
                for i in *
                do
                        grep "$2" $i #> /dev/null || echo $i
                done
        else
                for i in *
                do
                        grep "$1" $i #> /dev/null && echo $i
                done
fi
```

a. Describir qué hace.
b. Transcribir y probar en Linux.

8) Explique qué hace el siguiente Shellscript:

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

if [ $# -ne 1 ]; then
        echo "uso: $0 <patron> | -h"
        exit 1
fi

if [ $1 = "-h" ]; then
        echo "Texto explicativo."
        exit 0
fi

PATRON=$1

for i in `find ./ -name "*.[hc]"`
do
        echo "--------------"
        echo $i
        grep $PATRON $i
done
```