# Sistemas Operativos
# Tarea #1
## Shell Linux

Escuela de Ingeniería Civil Informática
Universidad de Valparaíso

Nombre: Joaquin Quintanilla Ramírez

## 1 Objetivo

|. El objetivo es familiarizarse con los comandos más usados y conocer algunas aspectos de la programación shell.

## 2 Instrucciones

Su trabajo deberá ser entregado en el servidor en su cuenta en GitHub.com. El nombre del repositorio debe ser SSOO-tarea01

```
+ /SSOO-tarea01
      - README.md
      - Makefile
      - src/
```

El archivo **`README.md`** es un archivo en markdown, donde se debe explicar el diseño de su solución, funciones utilizadas, etc. Además, debe especificar el nombre del autor y su correo institucional. En el directorio **`src/`** están los códigos fuentes de su proyecto. Además, debe de entregar un documento llamado "ApellidoPaterno_Nombre.pdf" evidenciando el trabajo realizado, de **NO** cumplir con el método de entrega solicitado será calificado con nota mínima.

## 3 Trabajo a realizar

a) Estudie y explique para qué sirven los comandos **ls**, **cat**, **chmod**, **echo**, **grep**, **cp**, **mv**, **rm** y **wc**. Dé ejemplos de uso. Para el comando **ls** averigüe para qué sirven las opciones -l, -t y -a. De ejemplo de uso para cada uno de esas opciones y la combinación de ellas. Ayuda: use el comando man, ej. **man ls**.

b) Explique qué son los metacaracteres y dé ejemplos de uso de ellos.

c) Explique en que conciste la expanción por paréntesis de conjunto (Brace Expansion). Con esta herramienta, resuelve el siguiente problema:

> En un directorio, se quieren crear subdirectorios para que almacenen respaldos diarios de todo un año. Debe tener la siguiente estructura :
>
> ```
> directorio_actual/
>       + 2021-01-01/
>       + 2021-01-02/
>       + 2021-01-03/
>       . . .
>       + 2021-09-18/
>       . . .
>       + 2021-10-18/
>       . . .
>       + 2021-12-31/
> ```
>
> Suponga que todos los meses tienen 31 días. Debe ejecutar UN sólo comando para crear la estructura de directorios solicitada.

d) La interconexión de comandos a través de *pipes* permite construir, de forma muy simple, nuevas herramientas. Como ejemplo considere los comandos ls y wc, que interconectados permiten contar archivos del directorio actual: ls | wc -l. Mediante el uso de pipes resuelva:

1. Mediante grep, encontrar archivos cuyo nombre contenga el caracter i en el directorio /bin.
2. Contar los archivos con una secuencia de permisos r-x en los directorios /bin y /usr/bin.

e) Las variables de ambiente definen aspectos del entorno de programación, y los comandos set y echo (mediante el metacaracter $) permiten ver su contenido.

1. Investigue el uso de las variables HOME, SHELL, PATH, y PWD. ¿Cómo se puede visualizar su contenido?
2. Investigue cómo se puede modificar el valor de una variable de ambiente (ayuda: investigue el operador '=' y export).
3. Ejecutando el comando *bash* dentro de la línea de comandos se crea un sub-shell, ¿Cómo afecta esto las variables de ambiente? ¿Cuál es el efecto de export? Explique y dé ejemplos

f) El intérprete de comandos bash es también un lenguaje de programación, con estamentos de control de flujo como for, while, etc. El código fuente a menudo se llama *script*. Si el archivo que contiene el script se llama **ejemplo.sh**, el comando **chmod +x ejemplo.sh**, le da permisos de ejecución al archivo **ejemplo.sh**.

Implementar un script BASH que liste cada argumento de entrada por separado, incluyendo el nombre del script. Además, debe mostrar su PID y mostrar las 10 primeras líneas del archivo /proc/PID/status.

# 4 Desarrollo

a)

- cat: El comando cat en Linux deriva su nombre de la palabra concatenar y este comando permite crear archivos, concadenarlos, redirigirlos o mostrar los archivos en la pantalla del terminal. Ejemplos de uso:
  - Creación de archivo: `cat > filename.txt`
  - Ver en pantalla: `cat filename.txt`
  - Redirigir: `cat source.txt > destination.txt`
  - Concadenar: `cat source1.txt source2.txt > destination.txt`

- chmod: En Linux y sistemas operativos de Unix el comando chmod sirve para cambiar los permisos de archivos o directorios, ósea, cambia el permiso de quien puede acceder al archivo o de cómo puede acceder a este. Ejemplo de uso:

  - La sintaxis del comando es primero el comando(chmod) después el usuario al que deseamos cambiar el permiso (u), el tipo de permiso que le queremos dar, que puede ser rwx (read-write-execute), rx (read-execute) o r(read) y por ultimo, el nombre del archivo(filename.txt)

    Código de ejemplo: `chmod u = rwx filename.txt`

- Echo: Este comando tiene la utilidad de imprimir en consola los argumentos definidos, pueden ser a modo indicativo, informativo o cualquier argumento que necesitemos visualizar. Ejemplo de uso:
  - `echo hola` (imprime en consola "hola")
  - `echo *jpg` (filtra por la extensión del archivo)

- grep: Este comando tiene la utilidad de buscar palabras o caracteres ingresados por nosotros en un archivo, el comando puede, o contar la cantidad de coincidencia, o indicar donde se encuentra esta palabra o carácter. Ejemplos de uso:
  - `grep palabra filname.txt` (busca "palabra" en el archivo "filename.txt")
  - `grep palabra *` (busca "palabra" en los archivos presentes en el directorio)
  - `grep -c palabra filename.txt` (cuenta las veces que se repite "palabra" en el archivo "filename.txt")

- cp: Este comando tiene la función de copiar un archivo o carpeta en donde nosotros indiquemos. Ejemplo de uso:
    - `cp filename.txt /home/user/desktop/Tarea01/` (copia "filename.txt" en el directorio ingresado)

- mv: Este comando tiene la utilidad de mover o renombrar archivos o directorios del sistema de archivos. Ejemplos de uso:
    - `mv filename.txt /home/user/desktop/Tarea01/` (mueve "filename.txt" en el directorio ingresado)
    - `mv filename1.txt filename2.txt` (cambia el nombre de "filename1" a "filename2")

- rm: Este comando tiene la función de eliminar archivos o directorios. Ejemplo de uso:
    - `rm filename.txt` (elimina el archivo "filename.txt". En caso de que dicho archivo este protegido se pedirá el permiso necesario para proseguir)

- wc: Este comando tiene la funcionalidad de contar tanto palabra como letras, bytes o líneas de un archivo. Ejemplo de uso:
    - `wc -m filename.txt` (imprime en consola el número de caracteres del archivo)

- ls: Este comando tiene la función de listar el contenido de un directorio, tanto del directorio en el que nos encontramos o de alguno que indiquemos. Ejemplos de uso:
    - `ls` (lista los archivos que hay en el directorio donde nos encontramos)
    - `ls ~/desktop/tarea01` (lista los archivos contenidos en ese directorio)

- ls -l: Esta opción nos permite listar los archivos en una línea por archivo con información adicional de cada uno, la información que se muestra de cada uno de estos es: permisos del fichero, el numero de enlace, nombre del propietario, nombre del grupo al que pertenece, tamaño de bytes, una marca de tiempo y nombre del fichero. Ejemplo de uso:
    - `ls -l filename.txt`

- ls -t: Esta opción ordena el listado de archivos a mostrar según la fecha de última modificación. Ejemplos de uso:
    - `ls -t filename.txt`
    - `ls-lt filename.txt`

- ls -a: Esta opción permite listar todos los ficheros, incluso los ocultos los cuales son marcados con un "." en el comienzo de su nombre. Ejemplo de uso:
    - `ls -a`

Estos comandos asociados a ls pueden ser combinados de la siguiente manera:

- `ls -lt`
- `ls -at`
- `ls -al`
- `ls -lta`

b) Los metacaracteres son caracteres no alfabéticos que poseen un significado especial en las expresiones normales, es decir, estos tienen una suerte de comodín en las expresiones, presentando un comportamiento diferente y variado en comparación a los caracteres tradicionales, cumpliendo la función de reemplazar o rellenar un espacio de una palabra o, por otro lado, remplazar uno o más caracteres y, según la propiedad del metacaracter, este provocaría que la palabra se tome de diferente forma.

Algunos de los metacaracteres más ocupados son:

- *: Se sustituye por cualquier secuencia de caracteres.
- ?: Se sustituye por cualquier carácter.
- [ ]: Se utiliza para especificar una lista de caracteres o un rango. Si se quiere especificar un rango se debe ubicar el carácter - entre el primer y el último carácter del rango. Se sustituye por un carácter de la lista o del rango.
- ^: Indica que la expresión comienza al principio de la línea.
- $: Indica el final de línea.
- * se refiere a cualquier conjunto de caracteres, incluido ningún carácter.
- z* cualquier conjunto de caracteres que empiece por z.
- *z cualquier conjunto que acabe en z.

Ejemplos:

- Cualquier fichero que empiece por la letra “i”:
  - `ls s*`
- Cualquier archivo que No empiece por la letra “i”:
  - `ls [!s]*`
- Muestra todos los archivos empiecen con letras o números pero que luego de ellos tengan los valores "ts.txt":
  - `ls ?ts.txt`
- Mostrará una lista de todos los archivos que cumplan con el rango de "a-f" sin importar la extensión o el nombre
  - `ls [a-f]*`
- Mostrará todos los archivos que no empiecen con una letra pero que sean extensión .txt:
  - `ls [^a-z].txt`

c) Brace Expansion en español expansión de llaves es una función que genera combinaciones entre los caracteres que se ingresen dentro de las llaves, el orden que usa es de izquierda a derecha, mediante la utilización de este método se puede llegar a facilitar el uso de un comando en reiteradas veces, ya que permite hacer iteraciones con diferentes caracteres en una sola definición del comando. Ejemplo:

- `echo a{1,2,3}` (Esto mostraría en el terminal a1 a2 a3 )
- también se puede ocupar definiendo rangos ejemplo : `echo {1..5}` (esto mostraría como salida 1 2 3 4 5 )
- Del mismo modo podemos establecer rangos pero con restricciones como por ejemplo que imprima los números del 0 al 10 pero de 2 en 2 ejemplo: `echo {0..10..2}` (esto mostraría como resultado en el terminal 0 2 4 6 8 10)
- Del mismo modo podemos usar lo anterior para crear múltiples carpetas ejemplo: `mkdir ~/desktop/tarea01/{uno,dos,tre,cuatro,cinco}` (esto crearía 5 carpetas con los respectivos nombres)
- Por ultimo y no mesnos importantes, se pueden hacer combinatorias de las expasiones ejemplo: `echo {a..c}{1..3}` (esto imprimiría en el terminal a1 a2 a3 b1 b2 b3 c1 c2 c3)

- Resolución de ejercicio

```
directorio_actual/
      + 2021-01-01/
      + 2021-01-02/
      + 2021-01-03/
      . . .
      + 2021-09-18/
      . . .
      + 2021-10-18/
      . . .
      + 2021-12-31/
```

Para crear los subdirectorios solicitados debemos en primer lugar definir el comando mkdir con el directorio actual mkdir ~/directorio_actual/. Luego de estos debemos usar la expansión de llaves para definir primero el año, un guion, los rangos de meses, otro guion, el rango de los días y por ultimo el /

Respuesta: `mkdir ~/directorio_actual/2021-{01..12}-{01..31}/`

d)

1) Para encontrar archivos los primero que debemos realizar es un ls en la ruta seleccionada, después de esto mediante un pipes hacemos un grep para buscar dentro de los nombres de los archivos alguno que contenga i, al grep lo usaremos con la opción -i para que busque tanto i mayúsculas como minúsculas.

- Resolución: `ls ~/bin | grep -i i`

2) Para contar los archivos con esos permisos primero que nada hacemos un ls a las dos rutas, esto se puede hacer escribiendo las dos rutas seguidas o como lo preferí hacer yo, que es con una expansión de llaves con las dos rutas para ahorrar espacio, el ls lo usamos con la opción -l para que quede en una fila las información y para que se visualice en primera posición los permisos; seguido de esto mediante una pipe hacemos un grep con el cual contaremos por fila el permiso r-x para que solo busque una coincidencia por fila y en nuestro caso cada fila va a corresponder a un archivo.

- Resolución: `ls -l ~/{bin,usr/bin} | grep -c r-x`

e) |1)

- HOME: es una variable de entorno que se usa mucho debido a que hace referencia a la carpeta personal del usuario. Para el usuario “user” la carpeta es: `/home/user`. Podemos visualizar su contenido con el comando echo y el metacarácter $ de la siguiente forma: `echo $HOME`.

- SHELL: Es una variable d entorno que contiene el nombre de la shell interactiva que se está ejecutando, por ejemplo bash. Podemos visualizar su contenido con el comando echo y el metacarácter $ de la siguiente forma: `echo $SHELL`.

- PATH: El sistema define algunas variables de forma predeterminada para nosotros, una de esas es la variable PATH, que contiene algunas rutas de ejecución importantes. Estas rutas son las predeterminadas que utiliza el shell para buscar un comando cuando se escribe en el terminal. Podemos visualizar su contenido con el comando echo y el metacarácter $ de la siguiente forma: `echo $PATH`.

- PWD: Es una variable de entrono la cual guarda el directorio actual de trabajo. Al igual que las demás para ver su contenido podemos ejecutar comando echo y el metacarácter $ de la siguiente forma: `echo $PWD`.

  2) Para modificar una variable de ambiente o entorno debemos emplear el comando `export variable = modificación`; por ejemplo si quisiéramos modificar la variable de zona horaria TZ deberíamos aplicar lo siguiente: `export TZ=”US/Pacific”`.