







LABORATORIO DE SISTEMAS (GIRS)

EXAMEN PRÁCTICO 1 - SHELL Y PYTHON SCRIPTING

15 DE MARZO DE 2023

-
-  **Profesores:** Elena García-Morato y Felipe Ortega.
 -  **Lenguaje de programación:** Bash, sed, awk, Python.
 -  **Hora tope de entrega:** Jueves, 16 de marzo de 2023 a las 10:40.
 -  **Envío:** Código fuente **comentado**, a través del espacio de entrega del examen en Aula Virtual.
-

INSTRUCCIONES

- Lee con detenimiento el enunciado de cada problema y plantea la solución antes de lanzarte a programar para resolverla.
- Puedes consultar el documento anexo con información de utilidad para resolver los ejercicios. No es posible consultar ninguna página de Internet ni acceder a los contenidos de Aula Virtual durante el examen.
- **Cada *script* debe estar almacenado en un fichero diferente. El nombre de cada fichero debe coincidir exactamente con el indicado en el enunciado del problema.**
- Una vez que termines, envía todos los *scripts* a través del espacio de entrega que se ha habilitado para el examen, en la sección Evaluación. No es necesario comprimir los archivos antes de enviarlos. Sube cada *script* al espacio de envío y luego confirma el envío para concluir el examen.
- **IMPORTANTE:** El espacio de envío se cierra automáticamente una vez superada la hora límite de finalización de la prueba. Por favor, no apures hasta el último momento para enviar tus respuestas.

Ejercicio 1

Escribe un script de shell llamado `permissions.sh` que se comporte de la siguiente manera:

- ▶ Si se ejecuta el *script* sin ninguna opción, imprime por pantalla una lista de los **archivos regulares** contenidos en el directorio actual o en alguno de sus subdirectorios (búsqueda recursiva), **incluyendo los archivos ocultos**, así como **información completa** sobre los mismos (propietario, permisos, tamaño, última modificación, etc.).
- ▶ Si se ejecuta con la opción `-x` seguida de una letra, concede **permisos de ejecución al usuario** para todos los **archivos regulares** contenidos en el directorio actual o en alguno de sus subdirectorios (búsqueda recursiva) cuyo nombre empiece por la letra indicada y cuya extensión sea `.txt`. En este caso no es necesario aplicar dicha modificación a los ficheros ocultos.
- ▶ Si recibe cualquier otra opción como parámetro o un número incorrecto de parámetros debe imprimir un mensaje de error alertando sobre ello y **finalizar devolviendo el código de status** apropiado.

Ejercicio 2

Escribe un *script* de shell llamado `processes.sh` que muestre por pantalla los diez procesos que consumen **más porcentaje de CPU**. A continuación, utiliza la **orden específica** de línea de comandos para averiguar el nombre del usuario que estás utilizando actualmente y almacena dicho dato en una variable. Por último, imprime por pantalla cuántos de estos procesos pertenecen a dicho usuario siguiendo el formato:

```
El usuario [nombre-usuario] es propietario de [N] de los
10 procesos que consumen más CPU actualmente.
```

Ejercicio 3

Escribe un *script* de shell con ayuda de los programas `sed` y `awk`, con el nombre `procesa_ls.sh`, que procese la salida del comando `ls -l` sobre un directorio cualquiera, produciendo el siguiente resultado:

- ▶ Solo se mostrarán las líneas correspondientes a archivos regulares. No se mostrarán directorios ni otros elementos.
- ▶ Se descarta la segunda columna (que muestra en la salida de `ls -l` el número de ítems contenidos en cada elemento). Esa columna **no debe aparecer** en la salida del *script*.
- ▶ La primera línea de la salida debe ser un título para el contenido de cada columna, por ejemplo:

PERMISOS	USUARIO	GRUPO	TAMAÑO	FECHA Y HORA	NOMBRE
----------	---------	-------	--------	--------------	--------

A continuación, se debe imprimir una línea en blanco para separar el título del resto del contenido.

- ▶ Se imprime el contenido de `ls -l`, quitando las líneas que no corresponden a ficheros regulares y la segunda columna de cada línea.
- ▶ Por último se debe imprimir una línea en blanco y, para acabar la salida del *script*, otra línea que indique el tamaño total de todos los archivos del directorio,
- ▶ Por ejemplo, si la salida del comando `ls -l` devuelve:

```
-rw-rw-r-- 1 jfelipe jfelipe 422 feb 20 16:25 ejemplo_parser.py
drwxrwxr-x 3 jfelipe jfelipe 4096 mar  9 00:09 gitrepo
-rwxrwxr-x 1 jfelipe jfelipe  42 feb 20 09:46 hello2.py
-rwxrwxr-x 1 jfelipe jfelipe  79 feb 20 09:50 hello3.py
-rwxrwxr-x 1 jfelipe jfelipe  46 feb 20 03:17 hello.py
```

La salida completa que produce el *script* debe ser:

PERMISOS	USUARIO	GRUPO	TAMAÑO	FECHA Y HORA	NOMBRE
-rw-rw-r--	jfelipe	jfelipe	422	feb 20 16:25	file1
-rwxrw-r--	jfelipe	jfelipe	42	feb 20 09:46	file2
-rwxrw-r--	jfelipe	jfelipe	79	feb 20 09:50	file3
-rwxrw-r--	jfelipe	jfelipe	46	feb 20 03:17	file4

TAMAÑO TOTAL: 589

Ejercicio 4

Crea un *script* en Python llamado `numeros.py` que se ajuste a las siguientes especificaciones:

- ▶ Como argumentos obligatorios debe recibir una serie de números enteros, separados entre sí por espacios en blanco. Como mínimo un solo número y como máximo 5 números.
- ▶ Si recibe el parámetro opcional `-h` o bien `--help` imprime información de ayuda sobre la sintaxis para ejecutar el *script* y las opciones disponibles.
- ▶ Si recibe el parámetro opcional `-m` o bien `--max` imprime el valor del máximo número incluido en la serie que hemos introducido. Ejemplo de ejecución:

```
$ python3 numeros.py --max 3 5 7 9
El máximo número es: 9
```

- ▶ Si recibe el parámetro opcional `-s` o bien `--min` imprime el valor del mínimo número incluido en la serie que hemos introducido. Ejemplo de ejecución:

```
$ python3 numeros.py --min 3 5 7 9
El máximo número es: 3
```

- ▶ Si recibe el parámetro opcional `-a` o bien `--asc` imprime la serie de números recibida ordenada de menor a mayor. Ejemplo de ejecución:

```
$ python3 numeros.py --min 5 3 9 7
La serie ordenada es: 3 5 7 9
```

- ▶ Si recibe el parámetro opcional `-r` o bien `--rev` imprime la serie de números recibida ordenada de mayor a menor. Ejemplo de ejecución:

```
$ python3 numeros.py --min 5 3 9 7
La serie inversamente ordenada es: 9 7 5 3
```