







LABORATORIO DE SISTEMAS (GIRS)

EXAMEN PRÁCTICO 2 - SHELL SCRIPTING, PYTHON SCRIPTING Y GIT

15 DE MAYO DE 2023

-
-  **Profesores:** Elena García-Morato y Felipe Ortega.
 -  **Lenguaje de programación:** Bash, sed, awk, Python, git.
 -  **Hora tope de entrega:** Lunes, 15 de mayo de 2023 a las 11:55.
 -  **Envío:** Código fuente **comentado** a través de Aula Virtual y GitLab de la EIF.
-

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 ficheros a través del espacio de entrega que se ha habilitado junto al examen, en la sección Evaluación. No es necesario comprimir los archivos antes de enviarlos. Una vez subidos al espacio de entrega, 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

El directorio `/etc/apt/sources.list.d` contiene una serie de archivos, todos ellos con extensión `.list`, que contienen información sobre repositorios software para instalar y actualizar aplicaciones en el sistema, utilizando el gestor de paquetes APT.

IMPORTANTE: Antes de comenzar el ejercicio, realiza una copia del directorio `/etc/apt/sources.list.d` (incluidos todos los archivos que contiene) en el directorio `/tmp`. Como resultado, las copias de todos los archivos con extensión `.list` quedarán accesible en la ruta `/tmp/sources.list.d`.

Crea un *script* de shell llamado `procesa-fuentes.sh`, utilizando las herramientas `sed` y `awk`, que se ajuste a las siguientes especificaciones de funcionamiento:

- ▶ Si no recibe ningún argumento, imprime por pantalla el número total de nombres de repositorios de fuentes software contenidos en todos los ficheros `*.list`. Cada repositorio se codifica en estos archivos con una línea que comienza por el identificador `deb` seguido de un espacio en blanco.
- ▶ Si recibe el parámetro `-u` o `--url`, entonces debe imprimir un listado con todas las direcciones URL de cada repositorio incluido en los ficheros `*.list`, de acuerdo con el formato (las etiquetas entre corchetes se deben rellenar con los datos correspondientes):

URLs

`https://cyberbotics.com/debian/`

`https://packages.gitlab.com/runner/gitlab-runner/ubuntu/`

`[resto de URLs]`

Total de URLs: `[número-total]`

- ▶ Si recibe el argumento `-a` o `--arch`, se debe imprimir el número total de repositorios codificados como arquitectura `i386` y el número total de repositorios con arquitectura `amd64`. El formato de salida será (las etiquetas entre corchetes se deben rellenar con los datos correspondientes):

ARQUITECTURAS

i386: [número-total]
amd64: [número-total]

- ▶ Si recibe cualquier otra opción como parámetro o un número incorrecto de parámetros debe imprimir un mensaje de error, alertando sobre ello, y finalizar con el código de status apropiado.

Ejercicio 2

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

- ▶ Debe recibir como argumentos obligatorios entre 1 y 4 números decimales, separados entre sí por un espacio en blanco.
- ▶ 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 `-f` o bien `--first` imprime el primer número decimal proporcionado por el usuario. Ejemplo de ejecución:

```
$ python3 decimales.py --first 3.5 6.7 2.0 8.5  
El primer número decimal es 3.5
```

- ▶ Si recibe el parámetro opcional `-p` o bien `--pi` comprueba si alguno de los números insertados es el número pi (3.14). Ejemplo de ejecución:

```
$ python3 decimales.py --pi 3.5 6.7 2.0 8.5  
La serie NO contiene el número pi
```

```
$ python3 decimales.py --pi 3.5 6.7 3.14 8.5  
La serie SI contiene el numero pi
```

- ▶ Si no se recibe ningún parámetro, realiza la suma de los números recibidos como argumento y la imprime por pantalla. Ejemplo de ejecución:

```
$ python3 decimales.py 3.5 6.7 2.0 8.5  
20.2
```

Ejercicio 3

En la dirección: `git@gitlab.etsit.urjc.es:egarciap/Madrid.git` se aloja un repositorio que contiene indicaciones para pasar un día haciendo turismo por Madrid. En él podrás encontrar dos archivos: `itinerario.txt`, que contiene un itinerario propuesto y `recomendaciones.txt`, con algunos consejos útiles para cualquier persona que visite la ciudad.

- ▶ En primer lugar, crea una copia de éste repositorio en tu máquina local y verifica el estado del mismo.
- ▶ Modifica el fichero `recomendaciones.txt` y añade una recomendación más, por ejemplo, *Llevar crema solar*. Registra un *commit* con dichos cambios descrito con el mensaje “*Nueva recomendación añadida*”.
- ▶ Partiendo de la única rama existente, crea una nueva rama llamada `museos`. Sobre ella, añade **al final del fichero `itinerario.txt`** una visita al Museo Del Prado a las 19:00. Vuelve a registrar los cambios en un nuevo *commit* con un mensaje descriptivo de tu elección.
- ▶ De vuelta en la rama `master`, añade **al final del fichero `itinerario.txt`** un lugar para cenar a las 21:00. Registra los cambios en un nuevo *commit* con un mensaje descriptivo.
- ▶ En realidad, nos parece muy interesante que cualquier persona que visite Madrid visite también el Museo del Prado, así que integra los cambios llevados a cabo en la rama `museos` sobre la rama principal, resolviendo los conflictos que pudieran generarse.
- ▶ Realiza un último *commit* que indique que ambas ramas han sido fusionadas y, a continuación, etiquétalo (con una etiqueta anotada) como la `v1.0`.
- ▶ Por último, desde la propia terminal de comandos, configura un repositorio remoto para tu repo local llamado `VisitaMadrid` en tu cuenta de GitLab de la EIF. Este segundo repositorio remoto se debe identificar en tu copia local por el alias `upstream`. Debes subir todos los cambios realizados y las etiquetas a este repositorio remoto para poder ser evaluado.

Envía a través del espacio de entrega de Aula Virtual un documento de texto plano nombrado `VisitaMadrid.txt` que contenga una línea con la URL *clonable* del repositorio remoto que has creado.