Trabajo Práctico Nro. 2 (GRUPAL):

- ***Tema:*** Programación de scripts en tecnología bash
- ***Descripción:*** Se programarán todos los scripts mencionados en el presente trabajo, teniendo especialmente en cuenta las recomendaciones sobre programación mencionadas en la introducción de este trabajo.
- ***Formato de entrega:*** Siguiendo el protocolo especificado anteriormente. Recomendamos realizar la entrega presencial
- ***Documentación:*** Todos los scripts que se entreguen deben tener un encabezado y un fin de archivo. Dentro del encabezado deben figurar el nombre del script, el trabajo práctico al que pertenece y el número de ejercicio dentro del trabajo práctico al que corresponde, el nombre de cada uno de los integrantes detallando nombre y apellido y el número de DNI de cada uno (tenga en cuenta que para pasar la nota final del trabajo práctico será usada dicha información, y no se le asignará la nota a ningún alumno que no figure en todos los archivos con todos sus datos), también deberá indicar el número de entrega a la que corresponde (entrega, primera reentrega, segunda reentrega, etc.).
- ***Evaluación:*** Luego de entregado el trabajo práctico los ayudantes procederán a evaluar los ejercicios resueltos, en caso de encontrar errores se documentará en la carátula del TP que será devuelta al grupo con la evaluación final del TP y una fecha de reentrega en caso de ser necesaria (en caso de no cumplir con dicha fecha de reentrega el trabajo práctico será desaprobado). Cada ayudante podrá determinar si un determinado grupo debe o no rendir coloquio sobre el trabajo práctico presentado.

  Las notas sobre los trabajos también estarán disponibles en el sitio de la cátedra (www.sisop.com.ar) donde además del estado de la corrección se podrá ver un detalle de las pruebas realizadas y los defectos encontrados

  Importante: Cada ejercicio cuenta con una lista de validaciones mínimas que se realizará, esto no implica que se puedan hacer otras validaciones al momento de evaluar el trabajo presentado.
- ***Fecha de entrega:*** **23/05/2018 o 24/05/2018**, dependiendo del día que se curse la materia.

Introducción:

La finalidad del presente práctico es que los alumnos adquieran un cierto entrenamiento sobre la programación de shell scripts, practicando el uso de utilitarios comunes provistos por los sistemas operativos (GNU/Linux). Todos los scripts, están orientados a la administración de una máquina, o red y pueden ser interrelacionados de tal manera de darles una funcionalidad real.
Cabe destacar que todos los scripts deben poder ser ejecutados en forma batch o interactiva, y que en ningún caso serán probados con el usuario root, por lo cual deben tener en cuenta al realizarlos, no intentar utilizar comandos, directorios, u otros recursos que solo están disponibles para dicho usuario.
Todos los scripts deberán funcionar en las instalaciones del laboratorio 266, ya que es ahí donde serán controlados por el grupo docente.
Para un correcto y uniforme funcionamiento, todos ellos deberán respetar algunos lineamientos generales indicados en el trabajo práctico anterior.

Ejercicios:

Tip: En caso de tener problemas con un script bajado desde un dispositivo formateado con DOS, y que contiene ^M al final de cada línea puede usar el siguiente comando para eliminarlos:

```
tr -d '\r' <archivo_con_M >archivo_sin_M
```

En caso de ejecutar scripts que usen el archivo de passwords, en el laboratorio, debe cambiar “cat /etc/passwd” por “getent passwd”

Ejercicio 1:

En base al siguiente script responda las preguntas que se detallan a continuación del mismo, tenga en cuenta que antes de poder ejecutarlo deberá otorgarle permisos de ejecución.

Importante: Se deberá entregar el script en un archivo con extensión .sh y las respuestas contenidas en el mismo como líneas comentadas.

```
#!/bin/bash

if [ $# != 1 ]; then
      echo "..."
      exit
fi
if ! [ -f "$1" ]; then
      echo "..."
      exit
fi
file -i "$1" | grep text/plain >> /dev/null

if [ $? != 0 ]
then
      echo "..."
      exit
fi

cp $1 $1.bak
ARCH1=/tmp/`basename $1`.$$
sed 's/[A-Z]*/\L&/g' $1 > $ARCH1
sed -i 's/^./\u&/' $ARCH1
mv $ARCH1 $1
```

Responda:

¿Qué significa la línea "#!/bin/bash"?

¿Con qué comando y de qué manera otorgó los permisos de ejecución al script?

¿Qué información brindan las variables $1, $? y $# ? ¿Qué otras variables similares existen? Explique de forma concisa el significado de cada una.

Explique el objetivo general del script.

Complete los echo "..." con mensajes acordes a cada situación según lo que se valida.

Explique de manera general la utilidad de sed y algunos parámetros.

Si ejecuto el script con un archivo con espacios en su nombre, da error. ¿Por qué? ¿Como lo corregiría?

Ejercicio 2:

Mejore el script del punto anterior, para que también corrija los espacios, tanto si hay dos o más seguidos como la falta de espacio luego de los signos de puntuación (punto, coma, punto y coma, etc… ojo con los punto y aparte).
También deberá recibir por parámetro un segundo archivo (pmayus.txt) que contendrá palabras que deben quedar tal cual están en ese archivo, por ejemplo las palabras que comienzan con mayúscula o son todas en mayúsculas, como nombres de países, provincias, nombre propios, siglas, etc.

Criterios de corrección:

Control	Criticidad
El script ofrece ayuda con –h, -? o –help explicando cómo se lo debe invocar	Obligatorio
Valida cantidad de parámetros mínimos.	Obligatorio
Funciona correctamente según enunciado.	Obligatorio
Funciona con archivos con espacio en su nombre.	Obligatorio
Se adjuntan archivos de prueba por parte del grupo.	Obligatorio

Funciona con rutas relativas y absolutas.	Obligatorio
Se implementan funciones	Opcional

Ejercicio 3:

Realizar un script que dado un ArchivoA con una lista de palabras separadas por saltos de líneas, encuentre la cantidad de apariciones de estas en un ArchivoB.
Deberá mostrar por pantalla la cantidad de coincidencias de cada palabra y la cantidad de palabras de B que no aparecen en el archivo A.
Las rutas hacia los archivos deberán ser enviadas como parámetros de forma obligatoria, debe manejar tanto rutas relativas como absolutas. Adicionalmente, se permitirá mandar un tercer parámetro (–i) para indicar que la búsqueda no diferenciará entre minúsculas y mayúscula. En caso de que este último parámetro se omita, la búsqueda deberá ser case sensitive.

Ejemplo de ejecución:
./miScript /path/ArchivoA /path/ArchivoB -i

ArchivoA	ArchivoB
hola mundo	hola hOLa mundo pepe Sistemas Hola operativos mundo

Salida por pantalla:
hola: 3
mundo:2
No existen en A: 3

Criterios de corrección:

Control	**Criticidad**
El script ofrece ayuda con –h, -? o –help explicando cómo se lo debe invocar	Obligatorio
Valida cantidad de parámetros mínimos.	Obligatorio
Funciona correctamente según enunciado.	Obligatorio
Funciona con directorios con espacios.	Obligatorio
Se adjuntan archivos de prueba por parte del grupo.	Obligatorio
Funciona con rutas relativas y absolutas.	Obligatorio
Se implementan funciones	Opcional

Ejercicio 4:

Se desea realizar un script para controlar el horario de trabajo de los empleados de la empresa.
Para esto se recibe un ReporteA, con nombre horario_yyyymm.log, que contiene los registros de las fichadas de cada trabajador en el mes, el mismo cuenta con el siguiente formato:

```
Legajo;dia;ingreso;egreso
001021;1;08:10:00;17:11:10
001023;1;08:05:20;17:00:36
001022;1;07:58:40;16:59:10
001023;2;08:00:20;17:10:36
……..
```

El script recibirá como parámetro, además del nombre del archivo a analizar (debe permitir rutas relativas y absolutas), alguno de los siguientes parámetros:

-r: Generara por cada trabajador un reporte de horas en otro archivo con nombre legajo_yyyymm.reg, con el siguiente formato:

```
Legajo;fecha;ingreso;egreso;horastrabajadas
001021;01/08/2017;08:10:00;17:11:10;09:01
```

…….
Total de horas teóricas: 184:00
Total de horas trabajadas: 190:21
Horas Extras: 6:00

La fecha en formato fecha (dd/mm/yyy) y el total de horas trabajadas serán solo horas y minutos, descartando segundos, luego de totalizar. Al final del archivo el resumen de horas, siendo horas teóricas: 22 días laborales por mes (lu-vi) * 8 hs. Y las horas extra la diferencia entre las teóricas y las trabajadas sin tener en cuenta los minutos.

-p: Va a continuación de -r únicamente, sino deberá informarse error de parámetros, y mostrará por pantalla los legajos procesados y cantidad de horas trabajadas de cada uno.

-l [legajo], Informara por pantalla las horas teóricas, trabajadas y extra de legajo pasado por parámetro. No debe generar el archivo de registro, sino calcularlo y mostrarlo.

Ejemplos de llamadas:
$ ./controlhorario.sh ./horario_201708.log -r
$ ./controlhorario.sh ./horario_201708.log -r -p
$ ./controlhorario.sh ./horario_201708.log -l 001021

Criterios de corrección:

Control	Criticidad
El script ofrece ayuda con –h, -? o –help explicando cómo se lo debe invocar	Obligatorio
Valida cantidad de parámetros mínimos.	Obligatorio
Funciona correctamente según enunciado.	Obligatorio
Funciona con directorios con espacios.	Obligatorio
Se adjuntan archivos de prueba por parte del grupo.	Obligatorio
Funciona con rutas relativas y absolutas.	Obligatorio
Se implementan funciones	Opcional

Ejercicio 5:

Realizar un script que reciba como parámetro la ruta y nombre de un archivo de log y permanezca en segundo plano capturando las señales SIGUSR1 y SIGUSR2, para las cuales se debe realizar las acciones que se describen a continuación, el resto de las señales deben ser ignoradas:

SIGUSR1: Realizar un backup comprimiendo todos los archivos y directorios que se encuentren dentro de la ruta indicada por la variable de entorno PATH_ENTRADA y guardar el archivo comprimido en el directorio indicado por la variable de entorno PATH_SALIDA. El nombre del archivo de salida debe ser el nombre del directorio concatenándole la fecha y hora al momento de la compresión. Además, deberá registrar en el fichero de log la fecha y hora de ejecución, la cantidad de archivos comprimidos y el peso del archivo final.

SIGUSR2: Elimina todos los archivos que se encuentren dentro de la variable de entorno PATH_SALIDA. Debe registrar en el fichero de log la fecha y hora de ejecución, cantidad de archivos eliminados y el espacio de disco liberado.

El script debe finalizar la ejecución al recibir la señal **SIGTERM**.
No debe permitirse que se ejecute más de una instancia de este script en todo el sistema.

Todas las rutas pueden ser relativas o absolutas.

Ejemplo de ejecución:
./ejecutar.sh registro.log

Criterios de corrección:

Control	Criticidad
El script ofrece ayuda con –h, -? o –help explicando cómo se lo debe invocar	Obligatorio
Valida cantidad de parámetros mínimos.	Obligatorio
Funciona correctamente según enunciado.	Obligatorio

Funciona con directorios con espacios.	Obligatorio
Se adjuntan archivos de prueba por parte del grupo.	Obligatorio
Funciona con rutas relativas y absolutas.	Obligatorio
Se implementa un script lanzador para la ejecución en segundo plano.	Obligatorio
Se implementan funciones	Opcional

Ejercicio 6:

Realizar un script que muestre los 10 subdirectorios más grandes de un directorio pasado por parámetro. Dichos subdirectorios no deben contener otros directorios dentro, o sea solo los directorios hoja en la rama pasada por parámetro. E informar la cantidad de archivos que contienen.
Ejemplo de salida:
/Oracle/client1/diag/Trace 509M 1500 arch.
/Oracle/client2/admin/log/Audit 6.1G 47689 arch.

Criterios de corrección:

Control	Criticidad
El script ofrece ayuda con –h, -? o –help explicando cómo se lo debe invocar	Obligatorio
Valida cantidad de parámetros mínimos.	Obligatorio
Funciona correctamente según enunciado.	Obligatorio
Maneja correctamente errores por permisos de acceso.	Obligatorio
Funciona con archivos con espacios en el nombre, cuando se utiliza -f.	Obligatorio
Se implementan funciones	Opcional