Actividad Práctica de Laboratorio Nro. 1:

- ***Tema:*** Programación de scripts en tecnología bash
- ***Descripción:*** Se programarán todos los scripts mencionados en el presente
- ***Formato de entrega:*** Siguiendo el protocolo especificado anteriormente
- ***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 número de APL al que pertenece y el número de ejercicio 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 de la APL 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 los docentes y ayudantes procederán a evaluar los ejercicios resueltos. Estas evaluaciones 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.

  Cada ayudante podrá determinar si un determinado grupo debe o no rendir coloquio sobre el contenido presentado.

  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.

- ***Planificación:***

Tipo	Fecha Inicial	Fecha final	Cursada presencial y semi-presencial	Cursada Virtual	Exámenes Libres
Validación	23/09/2020	24/09/2020	Obligatoria	Recomendada	No Aplica
Entrega Final	30/11/2020	01/12/2020	No Aplica	Obligatoria	No Aplica
Recuperatorio	14/12/2020	15/12/2020	Obligatoria	Obligatoria	No Aplica

- ***Introducción:***

  El objetivo pedagógico del presente 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, o en instancias similares de instalación y versionado de bibliotecas, ya que es ahí donde serán controlados por el grupo docente.

- ***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:

Tomando en cuenta el siguiente script responda las preguntas que se encuentran más abajo.
Importante: como parte del resultado se deberá entregar el script en un archivo tipo sh y las respuestas en el mismo código.

```
#!/bin/bash
ErrorS()
{
  echo "Error. La sintaxis del script es la siguiente:"
  echo ".....................: $0 nombre_archivo L" # COMPLETAR
  echo ".....................: $0 nombre_archivo C" # COMPLETAR
  echo ".....................: $0 nombre_archivo M" # COMPLETAR
}

ErrorP()
{
  echo "Error. nombre_archivo ....................." # COMPLETAR
}
if test $# -lt 2; then
  ErrorS
fi
if !test $1 -r; then
  ErrorP
elif test -f $1 && (test $2 = "L" || test $2 = "C" || test $2 = "M"); then
  if test $2 = "L" then
    res=`wc -l $1`
    echo "..............................: $res" # COMPLETAR
  elif test $2 = "C"; then
    res=`wc -m $1`
    echo "..............................: $res" # COMPLETAR
  elif test $2 = "M"; then
    res=`wc -L $1`
    echo "..............................: $res" # COMPLETAR
  fi
else
  ErrorS
Fi
```

Responda:

a. ¿Cuál es el objetivo de este script?
b. ¿Qué parámetros recibe?
c. Comentar el código según la funcionalidad (no describa los comandos, indique la lógica)
d. Completar los “echo” con el mensaje correspondiente.
e. ¿Qué información brinda la variable “$#”? ¿Qué otras variables similares conocen? Explíquelas.
f. Explique las diferencias entre los distintos tipos de comillas que se pueden utilizar en Shell scripts.
g. ¿Existe algún error en el código?, de ser así indique el motivo y corríjalo.

Ejercicio 2:

Se necesita realizar el análisis de un texto y para ello es necesario realizar un preprocesamiento que incluye una serie de transformaciones y normalizaciones. Este preprocesamiento incluye las siguientes etapas:

1. Eliminación de stop words
2. Convertir a mayúsculas

Los stop words serán reconocidos a través de un archivo de texto que se recibirá como parámetro y se eliminarán del texto a analizar. (https://en.wikipedia.org/wiki/Stop_words)
Una vez finalizado el preprocesamiento se deberá contabilizar la frecuencia de aparición de cada palabra en el texto y generar un reporte ordenando los resultados de mayor a menor frecuencia en un archivo de salida ***frecuencias_{Nombre archivo entrada}_{yyyy-mm-dd_hh:mm:ss}.out*** con formato CSV y por pantalla sólo se mostrarán las 5 palabras con mayor frecuencia.

Parámetros (el orden de los parámetros no debe ser fijo):

- -s “archivo stopwords”: path absoluto o relativo del archivo con los stopwords. Obligatorio.
- -o “directorio resultado”: path absoluto o relativo del directorio donde se generará el archivo de salida. Opcional. Si no se informa se generará en el directorio de ejecución.
- -i “archivo a analizar”: path absoluto o relativo del archivo de texto a analizar.

Resultado esperado:

1. Archivo de salida de frecuencias para el archivo analizado
2. Por consola se mostrarán las 5 palabras con mayor frecuencia absoluta

Por ejemplo:

COVID,20
CASOS,15
ARGENTINA,10
BRASIL,9
URUGUAY,5

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda (-h y --help)	Obligatorio
Validación correcta de los parámetros	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio
Utilización de sed para los reemplazos	Opcional

Ejercicio 3:

Para disminuir el espacio ocupado en un disco rígido, se desea determinar la existencia de archivos duplicados en el File System y también cuáles archivos superan en tamaño un umbral determinado. Para ello se necesita crear un script que recorra recursivamente un directorio y genere un informe sobre los archivos duplicados y los que superan el umbral.

Para el caso de archivos duplicados se deberá indicar la siguiente información:

- El nombre del archivo.
- Los directorios donde se encuentra dicho archivo.
- La fecha de última modificación de cada uno.

Para el caso de los archivos que superen el umbral de tamaño, se deberá indicar el path completo del archivo y el tamaño del mismo, ordenados de forma descendente.

Parámetros (el orden de los parámetros no debe ser fijo):

- -d “directorio”: path absoluto o relativo a analizar. Requerido.
- -o “directorio resultado”: path absoluto o relativo del directorio donde se generará el archivo de salida. Opcional. Si no se informa se generará en el directorio de ejecución.
- -u “umbral”: tamaño definido en KB para definir el umbral a analizar. Opcional. Si no es indicado, se considerará como umbral el promedio de peso de los archivos inspeccionados.

Resultado esperado:

1. Archivo de salida ***resultado_{yyyy-mm-dd_hh:mm:ss}.out*** indicando en primer lugar el resultado de los duplicados y luego el resultado de los archivos que superen el umbral.
2. Por consola se mostrará el mismo resultado generado en el archivo.

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda (-h y --help)	Obligatorio
Validación correcta de los parámetros	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio

Ejercicio 4:

Durante la cursada en un colegio secundario, se ha detectado que los alumnos utilizan las PCs del laboratorio para abrir programas que no corresponden al ámbito académico, en su gran mayoría juegos, por lo que se desea tener una herramienta para determinar cuándo se ejecutan determinados procesos que ya se han identificado. Para ello se necesita crear un script que se ejecute en segundo plano, que identifique cuándo se comienzan a ejecutar los procesos identificados en la lista negra de programas y los mate. Adicionalmente, se registrarán los eventos en un archivo de log, indicando la fecha y hora de ocurrencia, el ID del proceso, el nombre del proceso y el usuario que lo ejecutó.

Parámetros (el orden de los parámetros no debe ser fijo):

- -b “archivo”: path absoluto o relativo del archivo con la blacklist de procesos. Requerido.
- -o “directorio resultado”: path absoluto o relativo del directorio donde se generará el archivo de salida. Opcional. Si no se informa se generará en el directorio de ejecución.

Resultado esperado:

1. Archivo de salida ***blacklist_{yyyy-mm-dd_hh:mm:ss}.out*** con el formato indicado.
2. Al ser un proceso que se ejecuta en segundo plano, no se deberá mostrar nada por consola.

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda (-h y --help)	Obligatorio
Validación correcta de los parámetros	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio
No debe tener un alto consumo de CPU	Obligatorio

Ejercicio 5:

Se necesita mejorar la accesibilidad de una página web y para ello, el primer paso que se desea realizar es conocer el estado actual de la misma. Este proceso consiste en realizar un script que reconozca las etiquetas HTML utilizados en la página web que no contengan ningún atributo aria. Como no es necesario utilizar estos atributos en todas las etiquetas, se proporcionará por parámetro el archivo con las etiquetas que deberán ser analizadas. Para permitir que nuestro proceso sea flexible la lista de atributos aria será reconocida desde otro archivo de texto que se también se recibirá como parámetro. Una lista de ejemplos de atributos aria se puede encontrar en el siguiente link: https://www.w3.org/TR/using-aria/#aria-states-and-properties-aria-attributes
Una vez finalizado el proceso se deberá generar un archivo de texto en formato JSON con los resultados indicando: los tags que no están utilizando ningún atributo aria, la cantidad de veces que identificó en falta y los números de línea en donde se encuentran para su posterior corrección.

Parámetros (el orden de los parámetros no debe ser fijo):

- --aria: path absoluto o relativo del archivo que contiene la lista de atributos aria. Obligatorio.
- --tags: path absoluto o relativo del archivo que contiene la lista de etiquetas a analizar. Obligatorio.
- --web: path absoluto o relativo del archivo HTML a evaluar. Obligatorio.
- --out: path absoluto o relativo del archivo de salida. Obligatorio.

Resultado esperado:

1. Archivo de salida ***accessibilityTest_{yyyy-mm-dd_hh:mm:ss}.out*** con el formato indicado.
2. Se deberá mostrar por consola que el proceso finalizó y si es necesario o no realizar modificaciones en el archivo analizado.

Ejemplo del archivo de salida:

```
{
  [
    {
      "tag": "button",
      "cantidad": 4,
      "lineas": [2, 45, 77, 145]
    },
    {
      "tag": "img",
      "cantidad": 2,
      "lineas": [2, 45]
    }
  ]
}
```

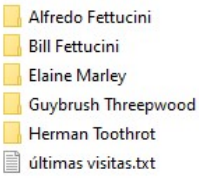
Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda (-h y --help)	Obligatorio
Validación correcta de los parámetros	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio

Ejercicio 6:

Un centro médico tiene en sus servidores discos rígidos de muy baja capacidad y que se encuentran casi llenos. Este problema, sumado al aumento de actividad a causa de la pandemia, llevó a los administradores del sistema de historias clínicas a tomar medidas urgentes para evitar el colapso del sistema, por lo que se decidió comprimir las historias clínicas de los pacientes que no visitaron el centro médico en los últimos días y eliminar los directorios con la información que está saturando al File System.

El directorio en donde se guardan las historias clínicas tiene el siguiente formato:



Cada uno de los directorios contiene la historia clínica de los pacientes que está compuesta por uno o más archivos (documentos, imágenes, etc).
El archivo “ultimas visitas.txt” contiene un listado de la última fecha en la que cada paciente visitó el centro médico. El formato del archivo es el siguiente: {Nombre Apellido}|{Fecha YYYY-MM-DD}

```
Alfredo Fettucini|2020-08-09
Elaine Marley|2020-06-05
Guybrush Threepwood|2020-06-05
```

Desarrollar un script que al ejecutarlo comprima las historias clínicas de los pacientes que no visitaron el centro médico en los últimos N días.

Parámetros (el orden de los parámetros no debe ser fijo):

- -c: comprimir. Si está presente, el script comprimirá las historias clínicas de los pacientes que no realizaron una visita en los últimos -n días, y eliminará los directorios para liberar el espacio. Obligatorio. No puede ser usado al mismo tiempo que -d.

- -n D: cantidad de días a tener en cuenta para comprimir las historias clínicas. Se puede utilizar solo con -c. Si no se indica se asumen 30 días.
- -d: descomprimir. Si está presente, el script descomprimirá la historia clínica del paciente indicado en el parámetro -p. Obligatorio. No puede ser usado al mismo tiempo que -c.
- -p “Nombre Paciente”: indica el nombre del paciente del cual se quiere descomprimir la historia clínica. Obligatorio. Solo puede ser usado si se usa el parámetro -d.
- -hc “directorio”: path relativo o absoluto del directorio en donde se encuentran las historias clínicas de los pacientes y el archivo “últimas visitas.txt”. Obligatorio.
- -z “directorio”: path relativo o absoluto del directorio en donde se guardan los archivos comprimidos. Obligatorio.

Resultado esperado:

1. Si se utiliza –c: Archivo/s comprimidos con las historias clínicas de los pacientes. Si se utiliza -d actualización del directorio de las historias clínicas.
2. Se deberá mostrar por consola si el resultado fue exitoso o no. En caso que se utilice –c se deberá indicar la cantidad de archivos que historias clínicas que se generaron.

Aclaraciones:

1. Se asume que “últimas visitas.txt” tiene siempre formato correcto, por lo que el script no debe validarlo. Si el archivo se encuentra vacío, se considera que tiene el formato correcto. Nunca aparecerán listados pacientes que no tienen una historia clínica.
2. Se debe validar la existencia del archivo “últimas visitas.txt”.
3. Los directorios con historias clínicas pueden no existir, ya que podrían haber sido comprimidos en una limpieza previa.

Criterios de corrección:

Control	Criticidad
Debe cumplir con el enunciado	Obligatorio
El script cuenta con una ayuda (-h y --help)	Obligatorio
Validación correcta de los parámetros	Obligatorio
Validación del archivo de entrada. Debe ser un archivo válido	Obligatorio
Proveer archivos de ejemplo para realizar pruebas	Obligatorio