**1. ¿Qué es el Shell Scripting? ¿A qué tipos de tareas están orientados los script? ¿Los scripts deben compilarse? ¿Por qué?**

Shell scripting

**2. Investigar la funcionalidad de los comandos echo y read**

**(a) ¿Como se indican los comentarios dentro de un script?**

Con #

**(b) ¿Cómo se declaran y se hace referencia a variables dentro de un script?**

let uno=1

nombre=”joaquin”

arreglo=(1 2 3)

${arreglo[1]}

**3. Crear dentro del directorio personal del usuario logueado un directorio llamado practicashell-script y dentro de él un archivo llamado mostrar.sh cuyo contenido sea el siguiente:**

**#!/bin/bash # Comentarios acerca de lo que hace el script # Siempre comento mis scripts, si no hoy lo hago # y mañana ya no me acuerdo de lo que quise hacer echo "Introduzca su nombre y apellido:" read nombre apellido echo "Fecha y hora actual:" date echo "Su apellido y nombre es: echo "$apellido $nombre" echo "Su usuario es: `whoami`" echo "Su directorio actual es:"**

**(a) Asignar al archivo creado los permisos necesarios de manera que pueda ejecutarlo**

**(b) Ejecutar el archivo creado de la siguiente manera: ./mostrar**

**(c) ¿Qué resultado visualiza?**

**(d) Las backquotes (`) entre el comando whoami ilustran el uso de la sustitución de comandos. ¿Qué significa esto?**

`comando`

$(comando)

**(e) Realizar modificaciones al script anteriormente creado de manera de poder mostrar distintos resultados (cuál es su directorio personal, el contenido de un directorio en particular, el espacio libre en disco, etc.). Pida que se introduzcan por teclado (entrada estándar) otros datos.**

**4. Parametrización: ¿Cómo se acceden a los parámetros enviados al script al momento de su invocación? ¿Qué información contienen las variables $#, $\*, $? Y $HOME dentro de un script?**

$1 primer parámetro y así

$# cant de param ;; $\* todos los param ;; $? Valor de retorno de la ultima función ;; $HOME dirección del /home del usuario

**5. ¿Cual es la funcionalidad de comando exit? ¿Qué valores recibe como parámetro y cual es su significado?**

Define el valor de retorno de una función o un script

Si el numero=0 el programa se ejecuto correctamente

Si el numero<>0 hubo un error

**6. El comando expr permite la evaluación de expresiones. Su sintaxis es: expr arg1 op arg2, donde arg1 y arg2 representan argumentos y op la operación de la expresión. Investigar que tipo de operaciones se pueden utilizar.**

+ - / \\* devuelve el resultado de la operación en la salida

< <= > >= devuelve en la salida el primer argumento si es verdad

**7. El comando “test expresión” permite evaluar expresiones y generar un valor de retorno, true o false. Este comando puede ser reemplazado por el uso de corchetes de la siguiente manera [ expresión ]. Investigar que tipo de expresiones pueden ser usadas con el comando test. Tenga en cuenta operaciones para: evaluación de archivos, evaluación de cadenas de caracteres y evaluaciones numéricas.**

**8. Estructuras de control. Investigue la sintaxis de las siguientes estructuras de control incluidas en shell scripting:**

**if**

**case**

**while**

**for**

**select**

**9. ¿Qué acciones realizan las sentencias break y continue dentro de un bucle? ¿Qué parámetros reciben?**

break sale de la iteración

continue sigue con la siguiente iteracion

**10. ¿Qué tipo de variables existen? ¿Es shell script fuertemente tipado? ¿Se pueden definir arreglos? ¿Cómo?**

Strings y arreglos

No es fuertemente tipado

**11. ¿Pueden definirse funciones dentro de un script? ¿Cómo? ¿Cómo se maneja el pasaje de parámetros de una función a la otra?**

Si

function nombre { block }

nombre() {block}

nombre 1 5 48

1, 5 y 48 son parámetros

**12. Evaluación de expresiones:**

**(a) Realizar un script que le solicite al usuario 2 números, los lea de la entrada Standard e imprima la multiplicación, suma, resta y cual es el mayor de los números leídos.**

**(b) Modificar el script creado en el inciso anterior para que los números sean recibidos como parámetros. El script debe controlar que los dos parámetros sean enviados.**

**(c) Realizar una calculadora que ejecute las 4 operaciones básicas: +, - ,\*, %. Esta calculadora debe funcionar recibiendo la operación y los números como parámetros**

**13. Uso de las estructuras de control:**

**(a) Realizar un script que visualice por pantalla los números del 1 al 100 así como sus cuadrados.**

**(b) Crear un script que muestre 3 opciones al usuario: Listar, DondeEstoy y QuienEsta. Según la opción elegida se le debe mostrar: Listar: lista el contenido del directoria actual. DondeEstoy: muestra el directorio donde me encuentro ubicado. QuienEsta: muestra los usuarios conectados al sistema.**

**(c) Crear un script que reciba como parámetro el nombre de un archivo e informe si el mismo existe o no, y en caso afirmativo indique si es un directorio o un archivo. En caso de que no exista el archivo/directorio cree un directorio con el nombre recibido como parámetro.**

**14. Renombrando Archivos: haga un script que renombre solo archivos de un directorio pasado como parametro agregandole una CADENA, contemplando las opciones:**

**“-a CADENA”: renombra el fichero concatenando CADENA al final del nombre del archivo**

**“-b CADENA”: renombra el fichero concantenado CADENA al principio del nombre del archivo**

**Ejemplo: Si tengo los siguientes archivos: /tmp/a /tmp/b**

**Al ejecutar: ./renombra /tmp/ -a EJ**

**Obtendré como resultado: /tmp/aEJ /tmp/bEJ**

**Y si ejecuto: ./renombra /tmp/ -b**

**EJ El resultado será: /tmp/EJa /tmp/EJb**

**15. Comando cut. El comando cut nos permite procesar la líneas de la entrada que reciba (archivo, entrada estándar, resultado de otro comando, etc) y cortar columnas o campos, siendo posible indicar cual es el delimitador de las mismas. Investigue los parámetros que puede recibir este comando y cite ejemplos de uso.**

**16. Realizar un script que reciba como parámetro una extensión y haga un reporte con 2 columnas, el nombre de usuario y la cantidad de archivos que posee con esa extensión. Se debe guardar el resultado en un archivo llamado reporte.txt**

**17. Escribir un script que al ejecutarse imprima en pantalla los nombre de los archivos que se encuentran en el directorio actual, intercambiando minúsculas por mayúsculas, además de eliminar la letra a (mayúscula o minúscula). Ejemplo, directorio actual:**

**IsO**

**pepE**

**Maria**

**Si ejecuto: ./ejercicio17**

**Obtendré como resultado:**

**iSo**

**PEPe**

**mRI**

**Ayuda: Investigar el comando tr**

**18. Crear un script que verifique cada 10 segundos si un usuario se ha loqueado en el sistema (el nombre del usuario será pasado por parámetro). Cuando el usuario finalmente se loguee, el programa deberá mostrar el mensaje ”Usuario XXX logueado en el sistema” y salir.**

**19. Escribir un Programa de “Menu de Comandos Amigable con el Usuario” llamado menu, el cual, al ser invocado, mostrará un menú con la selección para cada uno de los scripts creados en esta práctica. Las instrucciones de como proceder deben mostrarse junto con el menú. El menú deberá iniciarse y permanecer activo hasta que se seleccione Salir. Por ejemplo:**

**MENU DE COMANDOS**

**03. Ejercicio 3**

**12. Evaluar Expresiones**

**13. Probar estructuras de control**

**...**

**Ingrese la opción a ejecutar: 03**