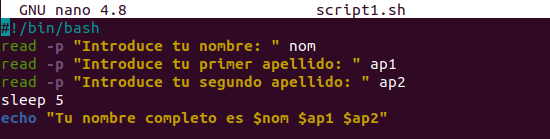
scripting en linux – ejercicios

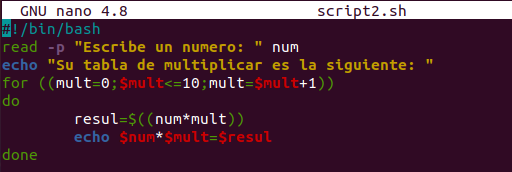
Para hacer estos ejercicios necesitamos una máquina virtual con Ubuntu 20

1. Crea un script que pida por teclado el nombre, después los dos apellido, permanezca esperando durante 5 segundos y después muestre por pantalla el mensaje "Tu nombre completo es …."



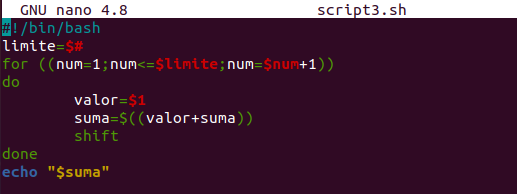
|  |
| --- |
| #!/bin/bash  read -p "Introduce tu nombre: " nom  read -p "Introduce tu primer apellido: " ap1  read -p "Introduce tu segundo apellido: " ap2  sleep 5  echo "Tu nombre completo es $nom $ap1 $ap2" |

1. Crea un script que pida un número y muestre su tabla de multiplicar.



|  |
| --- |
| #!/bin/bash  read -p "Escribe un numero: " num  echo "Su tabla de multiplicar es la siguiente: "  for ((mult=0;$mult<=10;mult=$mult+1))  do  resul=$((num\*mult))  echo $num\*$mult=$resul  done |

1. Crea un script que muestre la suma de todos los números que se le pasan como parámetro.



|  |
| --- |
| #!/bin/bash  limite=$#  for ((num=1;num<=$limite;num=$num+1))  do  valor=$1  suma=$((valor+suma))  shift  done  echo "$suma" |

1. Crea un script que muestre un menú con las opciones:
   * Suma
   * Resta
   * Multiplicación
   * División
   * Salir

Las 4 primeras opciones pedirán dos números y realizarán con esos números la operación correspondiente, mostrarán un mensaje con la solución (numero1 operación numero2 = resultado) y volverán a pedir que se elija una opción del menú (no hace falta volver a escribir el menú). La última opción terminará el script.

Utiliza funciones

|  |
| --- |
| [sudo] contraseña para sebas:  #!/bin/bash  pause(){  read -p "Presiona una tecla para continuar..." enterKey  }  opcion1(){  echo "Suma - Introduce un numero"  read num1  echo "Suma - Introduce otro"  read num2  resul=$((num1+num2))  echo "El resultado es $resul"  pause  }  opcion2(){  echo "Resta - Introduce un numero"  read num1  echo "Resta - Introduce un numero"  read num2  resul=$((num1-num2))  echo "El resultado es $resul"  pause  }  opcion3(){  echo "Multiplicacion - Introduce un numero"  read num1  echo "Multiplicacion - Introduce otro"  read num2  resul=$((num1\*num2))  echo "El resultado es $resul"  pause  }  opcion4(){  echo "Division - Introduce un numero"  read num1  echo "Division - Introduce otro"  read num2  if [ $num2 -eq 0 ]  then  echo "No es posible dividir entre 0"  else  resul=$((num1/num2))  echo "El resultado es $resul"  fi  pause  }  mostrarMenu(){  clear  echo "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"  echo " MENU - LINEA DE COMANDOS "  echo "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"  echo "1. Sumar"  echo "2. Restar"  echo "3. Multiplicar"  echo "4. Dividir"  echo "0. Salir"  }  leerOpcion(){  #Creo variable local  local opcion  #Leo la opcion y la guardo en la variable  read -p "Introduce una opcion [0-4]: " opcion  case $opcion in  1)opcion1;; #Lanzo la funcion opcion1  2)opcion2;; #Lanzo la funcion opcion2  3)opcion3;; #Lanzo la funcion opcion3  4)opcion4;; #Lanzo la funcion opcion4  0)exit 0;; #Salir del Script  \*)echo "Error: No se ha introducido un valor válido" && sleep 2 #Con cualquier otro numero  esac  }  while true  do  mostrarMenu #Muestro el menu  leerOpcion #Leo la opcion  done |

1. Crea un script para que haga las siguientes operaciones:

* Pedirá un nombre para los archivos que quieres crear
* Pedirá la cantidad de archivos que quieres crear
* Los archivos creados tendrán el siguiente formato (se crearán en el directorio personal del usuario:

*número\_de\_archivo\_nombre\_de\_archivo.txt*

* Dentro del archivo creado tendremos la siguiente frase:

Este es el archivo número (número de archivo)

|  |
| --- |
| #!/bin/bash  read -p "Introduce un nombre para los archivos: " nom  read -p "Introduce la cantidad de archivos: " num  for ((contador=1;contador<=num;contador=$contador+1))  do  archivo="$HOME/$nom""\_""$contador.txt"  echo "Este es el archivo $contador" > $archivo  echo "$archivo"  done |

1. Un administrador debe revisar día a día si hay conexión hacia todas las máquinas contenidas en un archivo de la forma:

maquina1;IP1

maquina2;IP2

…

Para hacerlo recuerda que el valor de retorno del comando ping es

* 0 🡪 hay conexión
* 1 🡪 la máquina destino no contesta
* 2 🡪 hubo algún error

El script cada vez que se ejecute añadirá a un archivo log (conexiones.log) con líneas similares a:

Conexiones – fecha\_del\_dia:

maquina1: esta viva

maquina2: no contesta

maquina3: error en ping

…

|  |
| --- |
| #!/bin/bash  fichero="/home/sebas/list.txt"  if test -f "$fichero"  then  while IFS= read -r linea  do  nombre=`echo $linea | cut -d "=" -f1`  ip=`echo $linea | cut -d "=" -f2`  ping -c1 $ip >/dev/null  respuesta1=$? #Guardamos estado  if [ $respuesta1 -eq 0 ];then  echo "$nombre: ESTA VIVA. HORA: $(date)" >> conexiones.log  elif [ $respuesta1 -eq 1 ];then  echo "$nombre: NO CONTESTA. HORA: $(date)" >> conexiones.log  else [ $respuesta1 -eq 2 ] then  echo "$nombre: ERROR EN PING. HORA: $(date)" >> conexiones.log  fi  done < $fichero  else  echo "El fichero no existe"  fi |

1. Crea un script que pide el nombre de un fichero y luego visualiza un menú con dos opciones: borrar fichero o visualizar contenido (tras realizar una de las acciones se muestra el resultado y se sale del script).

|  |
| --- |
| #!/bin/bash  pause(){  read -p "Presiona una tecla para continuar..." enterKey  }  opcion1(){  rm -r $fichero  echo "Fichero borrado con éxito"  pause  exit  }  opcion2(){  cat $fichero  pause  exit  }  mostrarMenu(){  clear  echo "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"  echo " MENU - LINEA DE COMANDOS "  echo "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"  echo "1. Borrar Fichero"  echo "2. Visualizar Datos del Fichero"  }  leerOpcion(){  #Creo variable local  local opcion  #Leo la opcion y la guardo en la variable  read -p "Introduce una opcion [1-2]: " opcion  case $opcion in  1)opcion1;; #Lanzo la funcion opcion1  2)opcion2;; #Lanzo la funcion opcion2  \*)echo "ERROR: Debes introducir uno de los valores indicados"  esac  }  while true  do  read -p "Introduce el nombre del fichero: " fichero  mostrarMenu #Muestro el menu  leerOpcion #Leo la opcion  done |

1. Crea un script que a partir de una lista de nombres de usuario contenida en el archivo alumnos.txt, uno por línea, determine si el usuario existe o no. El script mostrará un listado con los usuarios que figuran en el archivo alumnos.txt pero que no existan en el sistema.

|  |
| --- |
| #!/bin/bash  alumnos="/home/sebas/alumnos.txt"  if test -f "$alumnos"  then  echo "Alumnos que no existen en el sistema: "  while IFS= read -r linea  do  nombre=`echo $linea | cut -d ":" -f1`  getent passwd $nombre > /dev/null  if [ $? -ne 0 ]; then  echo $nombre  fi  done < $alumnos  else  echo "El fichero no existe"  fi |

1. Crea un script que indique los nombres de los diferentes shells asignados a las cuentas de usuario en el archivo /etc/passwd y la cantidad de cuentas de cada uno de esos shells

|  |
| --- |
| #!/bin/bash  dire="/etc/passwd"  if test -f "$dire"  then  while IFS= read -r linea  do  usuario=`echo $linea | cut -d ":" -f1`  nombre=`echo $linea | cut -d ":" -f7`  shell=`echo $SHELL`  echo "El usuario $usuario usa la shell $nombre"  done < $dire  bash=`cat /etc/passwd | cut -d ":" -f7 | uniq -c | sort`  echo $bash  else  echo "El fichero no existe"  fi |

1. Crea un script que pida un número entre 5 y 9 y en función de dicho número dibuje por pantalla un diamante como el de la siguiente figura (con tantas filas (para cada parte del diamante) como indica dicho número.

Por ejemplo, para el número 6 tendríamos la siguiente salida

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

\* \* \* \* \* \*

\* \* \* \* \*

\* \* \* \*

\* \* \*

\* \*

\*

|  |
| --- |
| #!/bin/bash  read -p "Dame un numero entre 5 y 9: " numlineas  if ! [ $numlineas -ge 5 -a $numlineas -le 9 ];then  echo "Ups... Debes ingresar un valor entre 5 y 9..."  fi  # Parte de arriba  for (( i=1; i<=numlineas; i++ ))  do  for (( s=numlineas;s>=i;s-- ))  do  echo -n " "  done  for (( j=1; j<=i; j++ ))  do  echo -n " \*"  done  echo ""  done  # Parte de abajo  for (( i=numlineas-1; i>=1; i--))  do  for (( s=i; s<=numlineas; s++ ))  do  echo -n " "  done  for (( j=1; j<=i; j++ ))  do  echo -n " \*"  done  echo ""  done |