BASH – PRÁCTICA 3
PROGRAMACIÓN SHELL-SCRIPTS
SOLUCIÓN PROPUESTA

Desarrollar dentro de un directorio, de nombre **Shell-Script,** cada uno de los siguientes programas. Cada programa se llamará ejercicio1.sh, ejercici2.sh, y así sucesivamente.

1. Programa que reciba tres palabras como parámetros y las visualice al revés.

```
#!/bin/bash
echo $3$2$1
```

2. Programa que visualice mensaje de error si no se le pasan parámetros.

```
#!/bin/bash
if [ $# -eq 0 ]
then
  echo "ERROR, No se han pasado parámetros"
fi
```

3. Programa que visualice un mensaje de error si no se le pasan parámetros, y que los visualice si se le pasan.

```
#!/bin/bash
if [ $# -eq 0 ]
then
  echo "ERROR, No se han pasado parámetros"
else
  echo "Los parámetros son:" $*
fi
```

4. Programa que reciba dos números y diga cuál es el mayor

```
#!/bin/bash
if [ $1 -gt $2 ]
then
  echo "El número $1 es mayor que el número $2"
else
  echo "El número $2 es mayor o igual al número $1"
fi
```

5. Programa que acepte dos números como parámetro y diga si el primero es mayor, menor o si son iguales.

```
#!/bin/bash

if [ $1 -gt $2 ]
then
  echo El número $1 es mayor que el número $2
elif [ $1 -eq $2 ]
then
  echo $1 es igual a $2
else
  echo El número $2 es mayor o igual al número $1
fi
```

6. Programa que reciba dos números como parámetro y devuelva la suma si el primero es mayor que el segundo y la resta en caso contrario.

```
#!/bin/bash

if [ $1 -gt $2 ]
then
  echo La suma es $(expr $1 + $2)
else
  echo La resta es $(expr $1 - $2)
fi
```

7. Programa llamado existe al que se le pasa como parámetro un fichero y dice si existe, no existe o es un directorio.

```
#!/bin/bash

if [ -f $1 ]
then
  echo El fichero $1 existe
elif [ -d $1 ]
then
  echo $1 es un directorio
else
  echo El fichero $1 no existe
fi
```

8. Realiza un shell-script que muestre todos los números pares, desde el 0 hasta el 100.

```
#!/bin/bash

for  ((num=0;num<=100;num+=2))
do
  echo $num
done
```


Francisco Javier Cabrerizo (fjavier.cabrerizo@outlook.es)

9. Realiza un shell-script, que calcule la potencia de un número. El shell pedirá la base y el exponente, devolviendo la potencia.

```
#!/bin/bash
echo "Introduce la base de la potencia"
read base
echo "Introduce el exponente de la potencia"
read exponente
potencia=1
for ((num=0;num<exponente;num++))
do
  potencia=$(( $potencia * $base)) 2>/dev/null
done
echo "La potencia es" $potencia
```

10. Realiza un shell-script que realice una división de dos números que se pasen por parámetro por el método de las restas.

```
#!/bin/bash
if [ $# -ne 2 ]
then
  exit 1
fi
dividendo=$1
divisor=$2
cociente=0
while [ $dividendo -ge $divisor ]
do
  dividendo=$(( $dividendo - $divisor ))2>/dev/null
  cociente=$(( $cociente + 1 )) 2>/dev/null
done
echo "El resultado de la división es:" $cociente
```

11. Implementar un shell-script que copie todos los ficheros que se le pasen por parámetro, al directorio Seguridad. Si directorio no existe, se deberá crear.

```
#!/bin/bash

if [ ! -d Seguridad ]
then
  mkdir -p Seguridad
fi
for fichero in $*
do
  if [ -f $fichero ]
  then
    cp $fichero Seguridad
  fi
done
```

12. Implementa un shell-scipt que indique si los ficheros pasados por parámetro existen y si son ficheros o directorios.

```
#!/bin/bash

for fichero in $*
do
  if [ -f $fichero ]
  then
    echo $fichero "es un fichero"
  elif [ -d $fichero ]
  then
    echo $fichero "es un directorio"
  else
    echo $fichero "no es ni fichero ni directorio"
  fi
done
```

13. Realiza un shell-script que muestre el nombre de cada uno de los ficheros pasados por parámetro y visualice su contenido por pantalla.

```
#!/bin/bash

for fichero in $*
do
  if [ -f $fichero ]
  then
    echo "Contenido del fichero" $fichero
    cat $fichero
  fi
done
```

14. Implementa un shell-script, que permita intercambiar el nombre de 2 ficheros pasados por parámetro.

```
#!/bin/bash

if [ $# -ne 2 ]
then
  exit 1
fi
mv $1 temporal.tmp
mv $2 $1
mv temporal.tmp $2
```

 Francisco Javier Cabrerizo (fjavier.cabrerizo@outlook.es)

15. Realiza un shell-script que limpie la pantalla, muestre todos los nombres de ficheros que hay en el directorio actual (mostrando sus atributos), la fecha actual y vuestro nombre.

```
#!/bin/bash

clear
ls -l
date
whoami
```

16. Realiza un shell-script que permita recibir, al menos, dos parámetros. El primero serán opciones, y el segundo un ficheros. Si el primer parámetro es -e, el shell ejecutará el programa que se pasa como segundo parámetro, si el primer parámetro es -m, el shell mostrará el contenido del fichero pasado como segundo parámetro.

```
#!/bin/bash

if [ $# -eq 2 ]
then
  case $1 in
    "-e")
      ./$2
      ;;
    "-m")
      cat $2
      ;;
    *)
      echo "Opción incorrecta"
      ;;
  esac
else
  echo "Número de argumentos incorrecto"
fi
```

17. Realiza un shell-script, que pida dos números por teclado. El programa pintará tantos *, como indique el número mayor de los dos introducidos.

```
#!/bin/bash

echo "Introduce el primer número"
read num1
echo "Introduce el segundo número"
read num2
if [ $num1 -gt $num2 ]
then
  mayor=$num1
else
  mayor=$num2
fi
for ((i=0;i<$mayor;i++))
do
  echo -n "*"
done
```

18. Implementa un shell-script que compruebe si los ficheros pasados por parámetro existen. Si existen, se muestra el nombre del fichero, si no existen, se muestra un mensaje de error.

```
#!/bin/bash
for fichero in $*
do
 if [ -f $fichero ]
 then
  echo "Existe el fichero" $fichero
 else
  echo "ERROR, no existe" $fichero
 fi
done
```