

U4 ADMINISTRACIÓN Y ASEGURAMIENTO DE LA INFORMACIÓN

ACTIVIDAD PRÁCTICA 14: Automatización (SCRIPTS)



24 de abril de 2023

SISTEMAS INFORMÁTICOS

Sergio Cobo García

Contenido

[Ejercicio 1 2](#_Toc133258904)

[Ejercicio 2 2](#_Toc133258905)

[Ejercicio 3 2](#_Toc133258906)

[Ejercicio 4 3](#_Toc133258907)

[Ejercicio 5 3](#_Toc133258908)

[Ejercicio 6 4](#_Toc133258909)

[Ejercicio 7 5](#_Toc133258910)

[Ejercicio 8 6](#_Toc133258911)

[Ejercicio 9 8](#_Toc133258912)

[Ejercicio 10 9](#_Toc133258913)

[Ejercicio 11 10](#_Toc133258914)

[Ejercicio 12 10](#_Toc133258915)

[Ejercicio 13 11](#_Toc133258916)

# Ejercicio 1

|  |  |
| --- | --- |
| 01 | El comando test dispone de un largo número de funciones que permiten hacer comparaciones. Llene las tres tablas siguientes, respectivas a los tres tipos de operadores de comparación de test. Indique qué valor tomaría la variable ?, en función de los valores introducidos (recuerde que 0 es verdadero en Linux). |

|  |  |
| --- | --- |
| Comparación numérica | |
| test n1 -eq n2 | ? valdría 0 si n1 es igual que n2 |
| test n1 -ge n2 | ? valdría 0 si n1 es mayor o igual que n2 |
| test n1 -gt n2 | ? valdría 0 si n1 es mayor que n2 |
| test n1 -le n2 | ? valdría 0 si n1 es menor o igual que n2 |
| test n1 -lt n2 | ? valdría 0 si n1 es menor que n2 |
| test n1 -ne n2 | ? valdría 0 si n1 no es igual que n2 |

|  |  |
| --- | --- |
| Comparación de cadenas de texto | |
| test s1 = s2 | ? valdría 0 si s1 es igual que s2 |
| test s1 != s2 | ? valdría 0 si s1 es no es igual que s2 |
| test -n s1 | ? valdría 0 si s1 es diferente a 0 |
| test -z s1 | ? valdría 0 si s1 es igual a 0 |

|  |  |
| --- | --- |
| Comparación de archivos | |
| test -d f1 | ? valdría 0 si f1 existe y es un directorio |
| test -e f1 | ? valdría 0 si f1 existe |
| test -f f1 | ? valdría 0 si f1 existe y es un archivo regular |
| test -r f1 | ? valdría 0 si f1 existe y tiene permisos de lectura |
| test -s f1 | ? valdría 0 si f1 existe y el tamaño es mayor que 0 |
| test -w f1 | ? valdría 0 si f1 existe y tiene permisos de escritura |
| test -x f1 | ? valdría 0 si f1 existe y tiene permisos de ejecución |
| test f1 -nt f2 | ? valdría 0 si la fecha de modificación de f1 es más reciente que la de f2 |
| test f1 -ot f2 | ? valdría 0 si la fecha de modificación de f1 es más antigua que la de f2 |

# Ejercicio 2

|  |  |
| --- | --- |
| 02 | Explique el funcionamiento de las comillas de este ejercicio, y por qué se utilizan. |

Echo “$comando”: ejecuta el valor de la variable comando, en este caso se lleva a cabo el comando ls.

Echo `$comando`: usa el resultado de la variable comando (ls).

Echo ‘$comando’: muestra literalmente el texto escrito entre las comillas simples.

# Ejercicio 3

|  |  |
| --- | --- |
| 03 | Indique justificadamente cuál es la función del siguiente shellscript, indicando cuál es el significado más lógico de los parámetros.  Escribid un comando equivalente a todo este shellscript. |

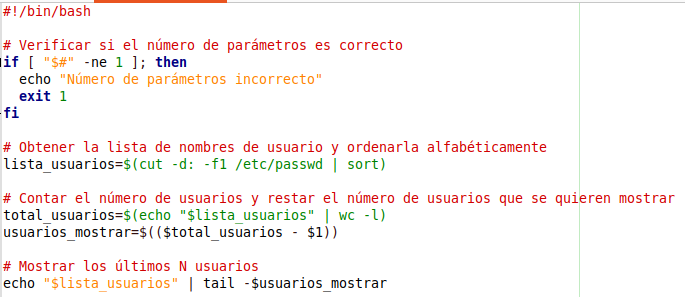
Se crea el archivo tmp. Seguidamente se inicia un bucle para que en cada archivo .txt busque la palabra “examen” y si la encuentra añade la línea que la contiene a tmp. A continuación, cuenta el número de líneas en el archivo tmp y lo imprime. Finalmente elimina el archivo tmp.

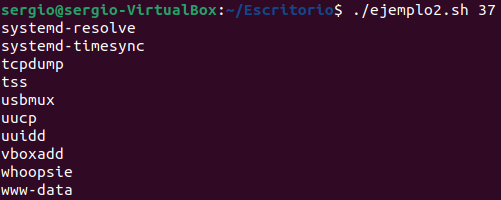
Un comando que hace lo mismo que el shellscript es el siguiente.



# Ejercicio 4

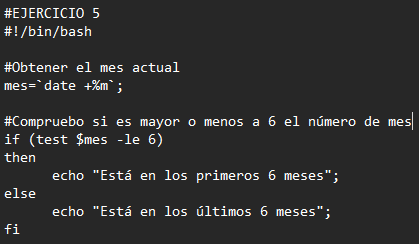
|  |  |
| --- | --- |
| 04 | Escriba un shellscript que a partir de un parámetro numérico N, ordene alfabéticamente los logins de usuario del sistema y muestre los N últimos. |





# Ejercicio 5

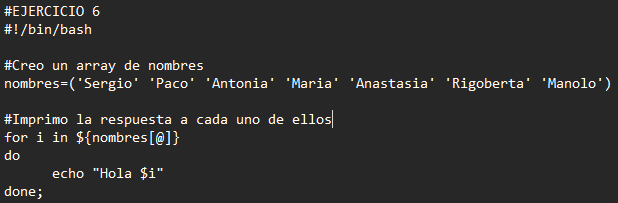
|  |  |
| --- | --- |
| 05 | Escriba un shellscript que indique si nos encontramos en los primeros o últimos seis meses del año. Tenga en cuenta que el sistema puede estar en cualquier idioma, por lo tanto, utilice los parámetros del comando "date" para obtener un valor válido para cualquier idioma. |

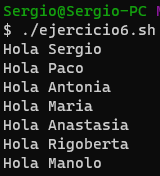




# Ejercicio 6

|  |  |
| --- | --- |
| 06 | Escriba un shellscript simple que, a partir de un número indeterminado de argumentos, salude a cada uno de los argumentos pasados. |

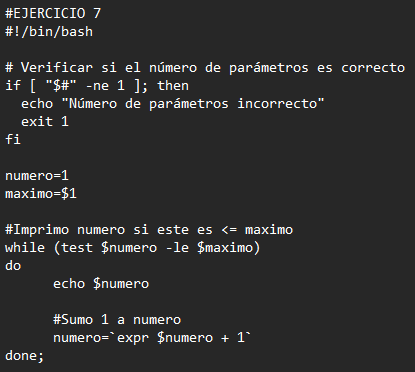


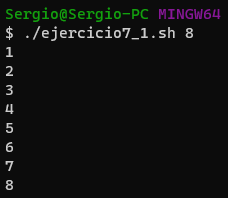


# Ejercicio 7

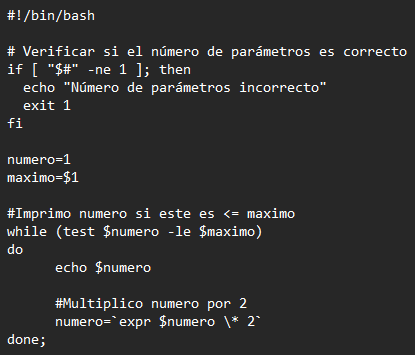
|  |  |
| --- | --- |
| 07 | Escriba un shellscript que, a partir de un único parámetro N, y utilizando el bucle "while", muestre por pantalla una progresión aritmética de N términos (1, 2, 3, 4...) y una progresión geométrica de N términos (1, 2, 4, 8, 16...). El número de términos de las sucesiones será el primer parámetro de este shellscript. |

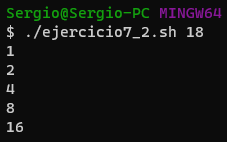
Progresión aritmética de N términos.





Progresión geométrica de N términos.

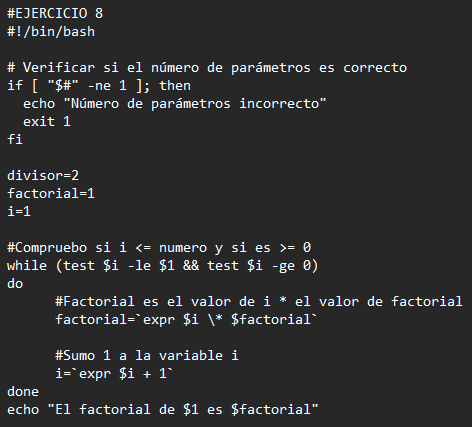


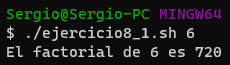


# Ejercicio 8

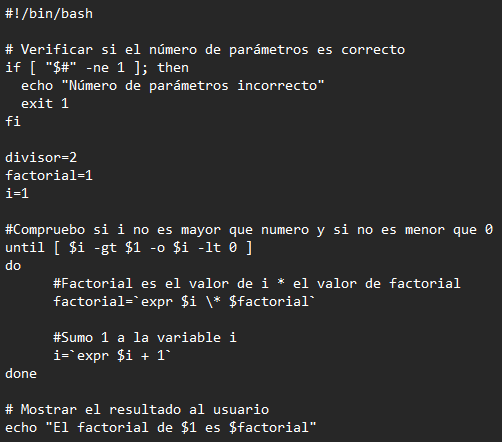
|  |  |
| --- | --- |
| 08 | Escriba un shellscript que, utilizando el bucle "while", muestre el factorial de un número por pantalla. A continuación, haga lo mismo con un bucle "until". El número sobre el cual calcular el factorial será el único parámetro de este shellscript ($1).  Recuerde que el factorial de un número es el producto de todos los números naturales desde 1 hasta ese número. Por ejemplo, el factorial de 6 (6!) será igual a 6 · 5 · 4 · 3 · 2 · 1. |

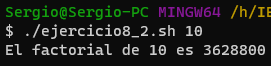
Con while





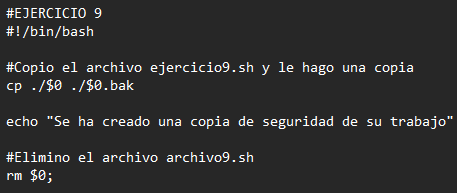
Con until

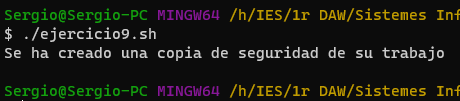




# Ejercicio 9

|  |  |
| --- | --- |
| 09 | Escriba un shellscript que intente autodestruirse (como archivo). Antes de ejecutar este shellscript, haga una copia de seguridad de su trabajo, por si acaso. No puede utilizar el nombre del archivo para eliminarlo. Comente el comportamiento del shellscript y si se borra o no el archivo. |

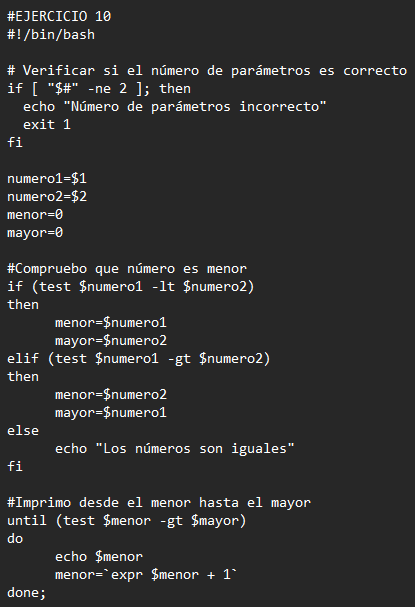


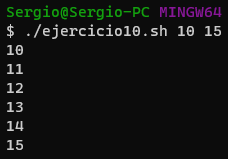




# Ejercicio 10

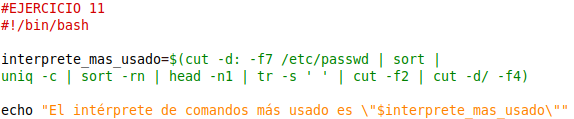
|  |  |
| --- | --- |
| 10 | Escriba un shellscript que, dados dos números diferentes pasados como parámetros, muestre por pantalla los números que van desde uno hasta el otro (ambos incluidos, crecientemente o decrecientemente). Debe resolver el ejercicio con un bucle until. |





# Ejercicio 11

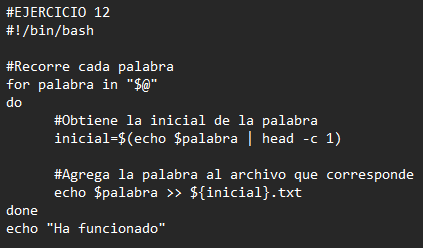
|  |  |
| --- | --- |
| 11 | Escriba un shellscript que, a partir del archivo /etc/passwd, diga cuál es el intérprete de comandos más utilizado por los usuarios del sistema (es decir, aquel que utilizan más usuarios). Este shellscript se puede resolver con una única (y larga) comando y sin utilizar ningún tipo de bucle. |





# Ejercicio 12

|  |  |
| --- | --- |
| 12 | Elabore un shellscript que reciba como parámetros un número indeterminado de palabras en minúscula. Lo que tendrá que hacer el shellscript será agregar cada una de las palabras a archivos que se llamarán como su inicial. |





# Ejercicio 13

|  |  |
| --- | --- |
| 13 | Indique justificadamente cuál es la función del siguiente shellscript, indicando cuál es el significado más lógico de los parámetros. |

El primer parámetro recibe la carpeta que contiene los archivos para recorrerlos, el segundo es la palabra que queremos buscar. Finalmente, si encuentra la palabra imprime A y lo copiará en el directorio /tmp sino imprime B.