Universidad de Valladolid – Departamento de Informática

Fundamentos de Informática II

Laboratorio - Programación del Shell de Unix

Guía de Prácticas

Sesión 3

Curso 2009/2010

En todos los casos a continuación si el script shell se realiza sin errores debe finalizar con un código de salida igual a cero. En otro caso debe finalizar con un código de salida distinto de cero.

1. Expresiones

1. Realizar un script que permita realizar cálculos aritméticos. El script debe solicitar los operadores y la operación a ser realizada, y luego presentar el resultado de la operación seleccionada. Por ejemplo.

```
Primer operador: 20
Segundo operador: 3
Operador ( +, -, *, /, % ) : %
El modulo de 20 y 3 es 2
```

2. Modificar el script anterior para que realice la operación varias veces, hasta que el primer operando sea un número negativo.

3. Realizar una nueva versión del primer script para que los tres datos se pasen como parámetros.

4. Realizar una nueva versión del anterior script, en la que si no se reciben exactamente tres argumentos se soliciten al usuario y se vuelvan a convertir en los argumentos del script.

5. Crea un guión que compruebe si su argumento es un número entero entre 0 y 100. Prueba distintas entradas, tanto cadenas que representen números o no.

2. Esquemas repetitivos

1. Realiza un guión que muestre por pantalla el tamaño en bytes de todos los ficheros ordinarios que estén en un directorio que se debe pasar como primer parámetro.

2. Realiza un guión que muestre por pantalla el número de subdirectorios de otro directorio que se pasa como primer argumento del guión.

3. Prepara un guión que reciba un número indeterminado de argumentos. Debe revisarlos, uno por uno, y sumar el número de líneas de todos aquellos que sean ficheros ordinarios. Al final, debe mostrar al usuario la suma total de las líneas.

4. Realizar un script que genere una serie de números fibonacci. Aceptar el parámetro n como argumento del script y comprobar su validez.

5. Realizar un script que genere una serie de números factoriales. Aceptar el parámetro n como argumento del script y comprobar su validez.

6. Escribir un guión que muestre por pantalla los tres primeros argumentos de un guión mediante un esquema `for`.

7. Realizar un script que muestre por pantalla todos sus argumentos mediante un esquema `for`:

   - De forma explícita
   - De forma implícita

8. Revisar el script anterior y justo después de mostrar el argumento actual aplicar un mandato `shift`. ¿Qué ocurre? ¿Por qué?

9. Realizar un script que permita comprobar si la cadena dada como argumento es palíndromo.

10. Realizar un script cuyo código de salida sea igual a la longitud total de las cadenas que se le hayan pasado como parámetro.

    - Realizar una versión usando el comando `expr`.
    - Realizar otra versión que no haga uso del comando `expr`.

11. Realizar un script que reciba un número arbitrario, no nulo, de argumentos y nos indique por pantalla cuáles de ellos son directorios.

12. Realizar un script que tenga un mínimo de dos argumentos. El primer argumento será el nombre de un directorio. El resto de argumentos serán nombres de ficheros. El script debe incluir el contenido de todos los ficheros en un nuevo fichero llamado `todos.txt`, siempre que sean ficheros ordinarios. `todos.txt` debe residir en el directorio destino que se ha pasado como primer parámetro. Realizar todas las validaciones necesarias para asegurar que el proceso se realizará sin incidencias.

13. Realizar un script que acepte un parámetro entero positivo impar. Se debe verificar la validez del dato. El script debe *dibujar* la siguiente figura de acuerdo al valor del parámetro.
Para n=3:

```
  *
 * *
* * *
```

Para n=5:

```
    *
   * *
  *   *
 *     *
* * * * *
```

Implementar tres versiones del script mediante el uso de bloques

- for
- while
- until

14. Realizar un script que acepte un parámetro entero positivo impar. Se debe verificar la validez del dato. El script debe *dibujar* la siguiente figura de acuerdo al valor del parámetro.
Para n=3:

```
* * *
*   *
* * *
```

Para n=5:

```
* * * * *
*       *
*       *
*       *
* * * * *
```

Implementar tres versiones del script mediante el uso de bloques

- for
- while
- until

15. Generar una variable que contenga una lista de cadenas, por ejemplo, los nombres de todos los ficheros o directorios del directorio actual. A continuación mostrar su contenido mediante un esquema `for`.

    - Modificar el script anterior, para que se muestren los ficheros o directorios de un directorio que se pasa como primer argumento.
    - Revisar el ejercicio anterior para que el script haga la comprobación de que dicho argumento existe y es un directorio. Si no existe, debe solicitarlo al usuario.
    - Generar una nueva versión del ejercicio anterior. En esta nueva versión deben listarse sólo los nombre de los ficheros ejecutables del directorio.
    - Proporciona otra versión en la que se muestren sólo los directorios, indicando su nombre completo: *ruta/nombre*.
    - Revisa nuevamente el ejercicio y ahora indica para cada uno de los archivos del directorio que se pasa como argumento, qué tipo de archivo es: fichero, directorio o de otro tipo.

16. Crear un script que reciba como argumentos cadenas de caracteres. Cada argumento indicará un nombre de usuario válido. El script debe mostrar los nombres reales de dichos usuarios, utilizando el comando `for` y el contenido del fichero `/etc/passwd`.

17. Crear un script que permita generar *skylines* artificiales, de tamaño n. El parámetro n debe pasarse al script como un argumento.
    Por ejemplo para n=*6* cualquiera de las siguientes salidas es válida:

```
n=6
***
****
******
**
****
******
```

```
n=6
***
***
******
******
**
****
```

    Se sugiere crear un script para generar números aleatorios, para simplificar el trabajo.