Archivos

Temas incluidos en la guía

- Archivos de texto.
- Archivos binarios.
- Diccionarios.

Ejercicios

> Nota: (★☆☆, ★★☆, ★★★) Esta notación indica la dificultad (ascendente).

Archivos de texto

1. ★☆☆ Escriba una función, llamada `less`, que reciba un archivo e imprima el contenido del archivo.

   > Observación: se puede resolver sin cargar todas las líneas del archivo en memoria.

2. ★☆☆ Escriba una función, llamada `head`, que reciba un archivo y un número `N` e imprima las primeras `N` líneas del archivo.

   > Observación: se puede resolver sin cargar todas las líneas del archivo en memoria.

3. ★☆☆ Escriba una función, llamada `tail`, que reciba un archivo y un número `N` e imprima las últimas `N` líneas del archivo.

   > Observación: se puede resolver sin cargar todas las líneas del archivo en memoria.

4. ★☆☆ Escribir una función, llamada `touch,` que reciba el nombre del archivo a guardar, y genere una archivo de texto vacío.

5. ★☆☆ Escribir una función, llamada `cp`, que copie todo el contenido de un archivo a otro, de modo que quede exactamente igual.

6. ★☆☆ Escribir una función, llamada `wc`, que dado un archivo de texto, lo procese e imprima por pantalla cuántas líneas, cuántas palabras, cuántos caracteres contiene el archivo, y el nombre del archivo.

   Ejemplo:

   ```
   > wc('file.txt')
   > 9 21 243 file.txt
   ```

7. ★★☆ Escribir una función, llamada `grep`, que reciba una cadena y un archivo, e imprima las líneas del archivo que contienen esa cadena.

8. ★★☆ Escribir una función, llamada `cat`, que reciba dos cadenas referidas a los nombres de dos archivos, y guarde en un tercer archivo el contenido de los dos archivos.

9. ★★☆ Escribir una función, llamada `rot13`, que reciba un archivo de texto de origen y uno de destino, de modo que para cada línea del archivo origen, se guarde una línea *cifrada* en el archivo destino. El algoritmo de cifrado a utilizar será muy sencillo: a cada caracter comprendido entre la a y la z (del alfabeto inglés), se le suma 13 y luego se aplica el módulo 26, para obtener un nuevo caracter.

10. ★★☆ Escribir una función `load_data` que reciba un nombre de archivo, cuyo contenido tiene el formato `key:value`, y devuelva un diccionario con el primer campo como clave y el segundo como diccionario.

11. ★★☆ Escribir una función `save_data` que reciba un diccionario y un nombre de archivo, y guarde el contenido del diccionario en el archivo, con el formato `key:value`.

12. ★★☆ Escribir una función, llamada `sed`, que reciba el nombre de un archivo, y dos cadenas. La primera cadena en el archivo es la que queremos reemplazar por la segunda.

13. ★★☆ Un archivo CSV (Comma Separated Value) es un archivo de texto que, en cada línea, almacena un registro. Cada registro está compuesto por campos. En el archivo, cada campo está separado por comas (`,`), de ahí el nombre. El siguiente texto es un ejemplo del contenido de un archivo con el mencionado formato:

    ```
    First Name,Last Name,Country,Date of Birth
    Emanuel,Ginobili,Argentina,1977
    Donald,Trump,USA,1946
    Carl,Gauss,Germany,1855
    ```

    Escribir una función que reciba la información que se quiere escribir en un archivo en formato CSV. Esta información tiene que estar estructurada en una secuencia de secuencias.

    Ejemplo:

    ```
    > info = [
      ["First Name","Last Name","Country","Date of Birth"],
      ["Emanuel","Ginobili","Argentina","1977"],
      ["Donald","Trump","USA","1946"],
      ["Carl","Gauss","Germany","1855"]
    ]
    ```

14. ★★☆ Un archivo TSV (Tab Separated Value) es un archivo, similar al CSV, que está escrito del siguiente modo:

    ```
    First Name  Last Name   Country Date of Birth
    Lewis   Hamilton    United Kingdom  1985
    Max Verstappen  Belgium 1997
    Charles Leclerc Monaco  1997
    ```

Donde cada espacio es un caracter tabular; en ASCII es el código 09 (distinto al caracter espacio).

Escribir una función que reciba la información que se quiere escribir en un archivo en formato TSV. Esta información tiene que estar estructurada en una secuencia de secuencias.

Ejemplo:

```
> info = [
  ["First Name","Last Name","Country","Date of Birth"],
  ["Lewis","Hamilton","United Kingdom","1985"],
  ["Max","Verstappen","Belgium","1997"],
  ["Charles","Leclerc","Monaco","1997"]
]
```

15. ★★☆ Dado el siguiente archivo de texto, que contiene un guion de la serie Seinfeld:

    1. Eliminar los nombres de los personajes.
    2. Eliminar lineas en blanco.
    3. Eliminar puntuaciones.
    4. Eliminar informacion adicional para los actores.
    5. Guardar el archivo con los cambios.

    Ejemplo:

    Si el archivo original es:

    ```
    jerry: (pointing at georges shirt) see, to me, that button is in the
    worst possible spot.

    george: are you through?

    jerry: you do of course try on, when you buy?
    ```

    El resultado tiene que ser:

    ```
    see to me that button is in the worst possible spot
    are you through
    you do of course try on when you buy
    ```

16. ★★★ Dado dos archivos en formato CSV, `dinosaurs1.csv` y `dinosaurs2.csv`, escribir un programa que lea los dos archivos guardados en disco, y luego imprima los nombres de los dinosaurios bípedos, desde el más lento al más rápido.

    `dinosaurs1.csv`

    ```
    ```

```
NAME,LEG_LENGTH,DIET
Hadrosaurus,1.4,herbivore
Struthiomimus,0.72,omnivore
Velociraptor,1.8,carnivore
Triceratops,0.47,herbivore
Euoplocephalus,2.6,herbivore
Stegosaurus,1.50,herbivore
Tyrannosaurus Rex,6.5,carnivore
```

dinosaurs2.csv

```
NAME,STRIDE_LENGTH,STANCE
Euoplocephalus,1.97,quadrupedal
Stegosaurus,1.70,quadrupedal
Tyrannosaurus Rex,4.76,bipedal
Hadrosaurus,1.3,bipedal
Deinonychus,1.11,bipedal
Struthiomimus,1.24,bipedal
Velociraptorr,2.62,bipedal
```

La velocidad se calcula como:

```
g = 9.8 # gravity constant
velocity = ((stride_length / leg_length) - 1) * sqrt(leg_length * g)
```

17. ★★★ Dado un archivo en formato CSV, que contiene canciones (nombre, duración, artista), escribir un programa que lea el archivo e imprima por pantalla la lista con el siguiente formato.

    Ejemplo:

```
+----------------------------+------+-----------------+
| Name                       | Time | Artist          |
+----------------------------+------+-----------------+
| Candy Shop                 | 3:45 | 50 Cent         |
| In Da Club                 | 3:13 | 50 Cent         |
| What Up Gangsta            | 2:58 | 50 Cent         |
| In The Night Of Wilderness | 5:26 | Blackmill       |
| Wicked                     | 2:53 | Future          |
| Cudi Montage               | 3:16 | KIDS SEE GHOSTS |
| Cellular                   | 2:58 | King Krule      |
| Pull Up                    | 3:35 | Playboi Carti   |
| The Birthday Party         | 4:45 | The 1975        |
| One                        | 4:36 | U2              |
+----------------------------+------+-----------------+
```

18. ★★★ Un archivo en formato JSON, es un archivo de texto, pero que su contenido está escrito del siguiente modo:

```
{
    "id": 22264,
    "name": "Jack",
    "last_name": "Hunt",
    "age": 38,
    "children": false,
    "siblings": "Jessica Hunt, Robin Hunt",
    "ssn": "1234-99-0012"
}
```

    Tiene el mismo formato que un diccionario de Python, pero esta escrito en un archivo de texto.

    Escribir una función que lea el archivo en formato JSON, y transforme el contenido a un diccionario de Python. Imprimir el diccionario por pantalla.

Archivos binarios

> Nota: es posible bajar archivos binarios de ejemplo para trabajar con los ejercicios en este link.

1. ★☆☆ Dado un archivo binario, leerlo y escribir el contenido por pantalla.
2. ★☆☆ Dado un archivo de texto, escribir una función que lea el archivo de texto y guarde el contenido en un archivo binario.
3. ★☆☆ Dado un archivo binario, leerlo y escribir el contenido por pantalla en formato de texto, es decir, que sea legible por un humano.
4. ★☆☆ Escribir un programa que genere un archivo binario grabando en él los números enteros del 0 al 1000000.