

18. Ficheros

1. Crea un módulo en Python que incluya una función diseñada por ti. Esta función debe devolver la ruta absoluta del directorio donde se encuentra el módulo que realiza la llamada.

Aunque al principio pueda parecer un poco confuso, esta función será muy útil en los próximos ejercicios, ya que te permitirá definir fácilmente la ubicación de los ficheros con los que trabajarás.

2. Crea un archivo con el nombre 1.txt y guárdalo en el mismo directorio en que vayas a guardar este ejercicio. El contenido del archivo ha de ser:

```
I, I will be king  
And you, you will be queen  
Though nothing, will drive them away  
We can beat them, just for one day  
We can be heroes, just for one day
```

3. Crea un módulo que pregunte por el título de la canción contenida en el archivo 1.txt del ejercicio anterior y por el nombre de su compositor.

A continuación, el programa debe crear una nueva versión del archivo inicial a la que asignaremos por nombre el título de la canción y como extensión .txt.

En esta nueva versión, la primera línea de texto ha de informarnos del título de la canción seguido del nombre del autor. A continuación, se mostrará la letra de la canción.

```
Título de la canción: xxxxxx  
Nombre del autor: yyyyyyyyyyyy  
xxxxxx - yyyyyyyyyyyy  
I, I will be king  
And you, you will be queen  
Though nothing, will drive them away  
We can beat them, just for one day  
We can be heroes, just for one day
```

4. Crea ahora un módulo que lea el contenido del archivo 1.txt y muestre por pantalla los caracteres entre el 10 y el 17 en una línea. A continuación se deben mostrar esos mismos caracteres en columna y mayúsculas (utilizar el método upper()).
5. Utilizando comandos Python, crea una copia del fichero 1.txt y asígnele el nombre 5.txt. A continuación, utiliza los métodos tell and seek para **sustituir los caracteres** comprendidos en este nuevo fichero entre las posiciones 10 y 17 por su traducción al español (no puedes escribir la línea completa, solamente sustituir la expresión indicada).
6. Desarrolla un módulo en Python que abra el archivo de texto 1.txt en modo lectura y procese su contenido de forma secuencial, leyendo una línea en cada iteración.

Para cada línea leída, el programa deberá:

- a) Mostrar la línea por pantalla.

- b) Solicitar al usuario la traducción de dicha línea al español.
- c) Escribir la traducción introducida en un nuevo archivo llamado traduccion.txt, respetando el orden original de las líneas.

El programa continuará este proceso hasta que se hayan leído y traducido todas las líneas del archivo de origen en el archivo de destino.

Para poder realizar los ejercicios 7, 8 y 9 debes descargar el archivo de datos auxiliares incluido en la tabla de ejercicios de la web del centro y descomprimir su contenido.

7. Para determinar la precisión de un arma de fuego se ha realizado el siguiente experimento:

Quinientos tiradores han disparado el arma una única vez, apuntando al centro de una diana. Las coordenadas de dicho centro son (4,4). En el fichero coordenadas.txt se han recogido las coordenadas de los quinientos disparos en el formato (x.xx,y.yy) (11 dígitos), siendo x.xx las coordenadas en el eje de las x e y.yy las coordenadas en el eje de las y. Ambas distancias son positivas y su valor máximo ha de ser menor de 10.

Abre el archivo coordenadas.txt y muestra por pantalla un listado en la que aparezcan las coordenadas de cada disparo en la forma "X = xx.x – Y = yy.y".

```
X = 1.67 - Y = 0.88
X = 2.69 - Y = 3.35
X = 3.88 - Y = 7.95
...
```

8. Partiremos de nuevo del fichero coordenadas.txt. Vamos a ir leyendo las coordenadas de cada uno de los disparos. Tras leer cada una de las coordenadas la pantalla ha de mostrar el valor de las coordenadas medias acumuladas incluyendo todos los disparos realizados hasta ese instante. ¿Qué puedes decir del funcionamiento de esta arma?

```
Disparo 1:   x = 4.81 y = 1.55      xmed = 4.810 ymed = 1.550
Disparo 2:   x = 7.40 y = 7.27      xmed = 6.105 ymed = 4.410
Disparo 3:   x = 5.89 y = 1.82      xmed = 6.033 ymed = 3.547
.....
```

```
# Truco para dar formato a los números decimales.
#Formateamos 5.1234554321
print "{0:.2f}".format(5.1234554321)
#El resultado sería
5.12
#¿Y si queremos tres?
print "{0:.3f}".format(5.1234554321)
#El resultado sería
5.123
```

9. Modifica el programa anterior para que tras cada disparo también se muestre la distancia media a la diana:

Distancia entre dos puntos:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

```
Disparo 1:   x= 4.81 y= 1.55      xmed= 4.810 ymed= 1.550   Error:
2.580
```

```
Disparo 2:      x= 7.40 y= 7.27      xmed= 6.105 ymed= 4.410  Error:
2.145
Disparo 3:      x= 5.89 y= 1.82      xmed= 6.033 ymed= 3.547  Error:
2.083
...
```

10. Crea un módulo que pregunte a un alumno por la calificación obtenida en cada una de las siguientes materias: Matemáticas, Lengua, Historia y TIC II. Las calificaciones serán números enteros. El resultado se debe ir guardando en un diccionario de nombre "notas". Cada elemento del diccionario tendrá como índice el nombre de la materia y como valor asociado la calificación conseguida por el alumno en dicha materia. Muestra el contenido del diccionario notas y la nota media de todas las materias.
11. Modifica el programa anterior. Utilizando el **módulo pickle** crea el archivo notas.pkl. Este archivo ha de guardar la información relativa al nombre de las asignaturas y sus calificaciones.
12. Utilizando el **módulo pickle** abre el archivo notas.pkl. Lee la información contenida en el mismo y muestra por pantalla de forma tabulada la información contenida en él.

```
Matemáticas      -> 10
Lengua           -> 9
Historia          -> 5
TIC II           -> 7
Calificación media obtenida: 7.75
```

13. En este ejercicio vamos a crear un encriptador de mensajes. El método de encriptación consistirá en sumar un valor entero fijo al código ASCII de cada carácter del texto original. El programa deberá solicitar al usuario un texto mediante input. A continuación, pedirá la clave de encriptación, que será un número entero que se sumará al código ASCII de cada carácter del texto a encriptar.

Por último, el programa mostrará por pantalla el texto encriptado resultante.

chr(num): Devuelve el carácter asociado al valor "num" en código ASCII.
ord('x'): Devuelve el código ASCII del carácter 'x'.

```
Introduce el texto a encriptar: En un lugar de la marcha de cuyo
nombre no me quiero acordar
Introduce la clave de encriptación: 5
Js%zs%qzlfw%ij%qf%rfwhmf%ij%hz~t%strgwj%st%rj%vznjwtfhtwifw
```

14. Modifica el programa anterior. Utilizando el **módulo pickle** el programa ha de guardar el mensaje encriptado. Sólo quien conozca la clave de encriptación podrá leer el mensaje.
15. Crea un programa que conociendo la clave de encriptación lea cualquier archivo creado utilizando el programa del ejercicio anterior.

```
Texto a desencriptar
Hshu'[|ypun'klzjpmý'sh'tèx|puh'Lupnth5
Introduce la clave de desencriptación: 7
Texto desencriptado:
Alan Turing descifró la máquina Enigma.
```

16. Crea un programa que, utilizando el módulo `Shelve`, gestione una base de datos con los alumnos de un centro educativo.

El programa deberá **solicitar los datos de los alumnos de forma interactiva** y seguir pidiéndolos hasta que el usuario indique que desea finalizar.

De cada alumno se almacenarán los siguientes datos: **código de matrícula, nombre, dirección y curso**. El **código del alumno** se utilizará como clave principal del almacén.

La información de cada alumno se guardará en forma de **diccionario** y, cada vez que se introduzcan los datos de un alumno, estos deberán **guardarse inmediatamente en el almacén shelve**.

17. Crea un programa que **abra la base de datos del ejercicio anterior** y recorra todos los alumnos almacenados.

El programa deberá **mostrar por pantalla los nombres completos de los alumnos**.

18. Tomando como punto de partida el ejercicio 4 de la hoja 14 (Módulo `random`) modifica el código del programa para que utilizando el **módulo shelve** se guarden en un archivo independiente los datos de las tres mejores puntuaciones conseguidas en el juego, así como el nombre de los jugadores.

- Esta información se debe mostrar al comienzo de la partida y al finalizar la misma, ya actualizada, si el jugador ha conseguido entrar en la lista.
- Te recomiendo crear un módulo independiente que genere el archivo inicial con las tres puntuaciones vacías.