PYTHON PARA LINGÜISTAS

LECTURA Y ESCRITURA DE FICHEROS

CSV Y XML

ALEJANDRO ARIZA

CENTRE DE LLENGUATGE I COMPUTACIÓ

UNIVERSITAT DE BARCELONA

¿QUÉ SABEMOS HASTA AHORA?

- Lectura de ficheros (with open... as f:).
- Procesado de ficheros línea por línea usando listas y strings.
- Escritura de ficheros línea por línea.

TRABAJAR CON FICHEROS MÁS COMPLEJOS

- Trabajar con ficheros de texto generalmente es mejor que trabajar con variables.
- A veces, necesitamos trabajar con tipos de ficheros más complejos (con un formato determinado).
- Comenzando con esta clase, aprenderemos a cómo usar ciertas herramientas en Python (librerías).

LECTURA Y ESCRITURA DE FICHEROS

```
with open('macbeth.txt', 'r', encoding='utf-8') as f:
    with open('copy.txt', 'w', encoding='utf-8') as g:
        for line in f:
            print(line, file=g)
```

- Con este código podemos:
  - Leer ficheros de texto sin importar su contenido.
  - Escribir en ficheros de texto con el formato que deseemos.

CSV: COMMA SEPARATED VALUE

- Formato similar a Excel donde las columnas están separadas por un carácter especial (coma, punto y coma...)

  id;pelicula;año;páginas

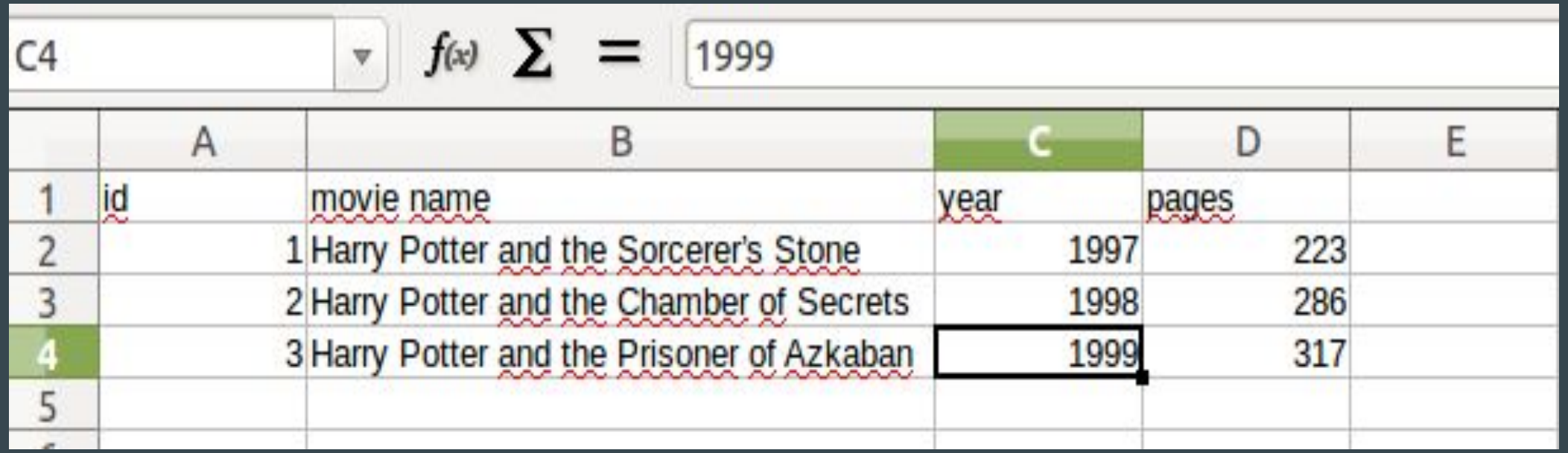
  1;Harry Potter y la Piedra Filosofal;1997;223

  2;Harry Potter y la Cámara de los Secretos;1998;286

  3;Harry Potter y el Prisionero de Azkaban;1999;317

CSV (2)

- Un fichero CSV puede ser importado directamente en Excel:



¿QUÉ ESTRUCTURA DE DATOS USAMOS PARA CSV?

- String: “1;Harry Potter y la Piedra Filosofal;1997;223”
- Lista: [1, ”Harry Potter y la Piedra Filosofal”, 1997, 223]
- Diccionario: {“id”: 1,
  “titulo”: “Harry Potter and the Sorcerer’s Stone”,
  “año”: 1997,
  “páginas”: 223}

EL MÓDULO CSV

- Para leer ficheros .csv, usaremos el módulo csv.
- Los módulos son recursos externos (incluyendo funciones y tipos de datos) que no están disponibles en el Python “básico”. Los utilizaremos para añadir funcionalidades a nuestro programa.
- Los módulos se introducen con la palabra reservada “import”
- Importaremos el módulo csv de la siguiente forma:

```
import csv
```

LECTURA DE FICHEROS CSV

```
import csv
with open('movie_plots.csv', 'r') as f:
    reader = csv.reader(f)
    for row in reader:
        print(row[1])
```

- Es muy similar a lo que ya conocemos pero, en vez de leer un string por cada línea, lee una lista de elementos.
- Si el fichero .csv contiene el nombre de las columnas en la primera línea, ésta debe ser tratada de forma diferente.

LECTURA DE FICHEROS CSV (2)

```
import csv
with open('movie_plots.csv', 'r') as f:
    reader = csv.DictReader(f)
    for row in reader:
        print(row['Title'])
```

- Si utilizamos DictReader en vez de reader(), obtendremos un diccionario en lugar de una lista.
- 1ª fila = nombre de columnas (por defecto)

ESCRITURA DE FICHEROS CSV

```
import csv
with open('potter.csv', 'w') as f:
    writer = csv.writer(f)
    writer.writerow(mylist)
```

- writerow(List) escribe una fila, usando como nombres de columna los elementos de la lista.

ESCRITURA DE FICHEROS CSV

```
import csv
with open('potter.csv', 'w') as f:
    writer = csv.DictWriter(f)
    writer.writerow(mydict)
```

- En este caso, writerow(Dict) toma un diccionario y rellena los valores de las columnas utilizando los nombres de las columnas como las claves.

PROCESAR CSV. MÁS OPCIONES.

- El delimitador por defecto es la coma.

  Aunque podéis especificar otros delimitadores tanto al leer como al escribir:
  csv.reader(f, delimiter=';')

- Existen muchas funciones que podéis encontrar en su documentación:

  https://docs.python.org/3/library/csv.html

- En la práctica, podemos considerar otras librerías como Pandas con funciones como read_csv() y to_csv()

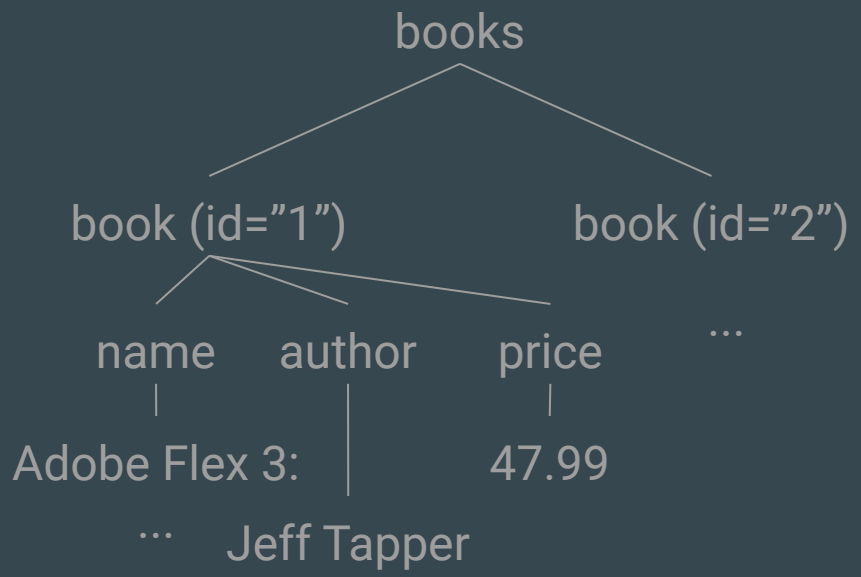
XML
<TAG>EXTENSIBLE MARKUP LANGUAGE</TAG>

```
<!-- This is a sample XML file for various XML example-->
<books>
  <book ID="1">
    <name>Adobe Flex 3: Training from the Source</name>
    <author>Jeff Tapper</author>
    <price>47.99</price>
    <type>Flex</type>
    <image>AdobeFlex3.jpeg</image>
  </book>
  <book ID="2">
    <name>Styling Web Pages with CSS</name>
    <author>Tom Negrino</author>
    <price>15.99</price>
    <type>CSS</type>
    <image>CSS.jpeg</image>
  </book>
```

FORMATO XML

```xml
<!-- This is a sample XML file for various XML example-->
<books>
  <book ID="1">
    <name>Adobe Flex 3: Training from the Source</name>
    <author>Jeff Tapper</author>
    <price>47.99</price>
    <type>Flex</type>
    <image>AdobeFlex3.jpeg</image>
  </book>
  <book ID="2">
    <name>Styling Web Pages with CSS</name>
    <author>Tom Negrino</author>
    <price>15.99</price>
    <type>CSS</type>
    <image>CSS.jpeg</image>
  </book>
```

XML → FORMA DE ÁRBOL (JERARQUÍA)




```
<!-- This is a sample XML file for various XML example-->
<books>
  <book ID="1">
    <name>Adobe Flex 3: Training from the Source</name>
    <author>Jeff Tapper</author>
    <price>47.99</price>
    <type>Flex</type>
    <image>AdobeFlex3.jpeg</image>
  </book>
  <book ID="2">
    <name>Styling Web Pages with CSS</name>
    <author>Tom Negrino</author>
    <price>15.99</price>
    <type>CSS</type>
    <image>CSS.jpeg</image>
  </book>
```

TRABAJAR CON XML

- La librería básica para procesar XML en Python se llama ElementTree
- Podéis importarla de la siguiente forma:

  ```
  import xml.etree.ElementTree as ET
  ```

- Podéis leer un fichero .xml usando la siguiente sentencia:

  ```
  tree = ET.parse('potter.xml')
  ```

- Para más información acerca de las herramientos de esta librería, podéis mirar su documentación:

  https://docs.python.org/3/library/xml.etree.elementtree.html

PROCESAR FICHEROS EN PYTHON

- Ficheros de texto en Python:
  - Open/Close: with open... as f
  - Read: for line in f
  - Write: print(), f.write()
- Lectores de ficheros CSV y XML:
  - Cuando tu problema es “común”, alguien ya lo ha resuelto y publicado una librería. Impórtala.
- Existen muchas otras herramientas para otros formatos de ficheros:
  - JSON https://docs.python.org/3/library/json.html
  - Excel https://pypi.org/project/xlrd/

¡GRACIAS!
¿PREGUNTAS?