Clase4_Archivos

April 3, 2023

0.0.1 Seminario de Lenguajes - Python

0.1 Cursada 2023

0.0.1 Archivos. Formatos JSON y CSV

1 Les dejé algo para investigar en el video del fin de semana....

1.1 ¿Cuándo un módulo se denomina ___main___?

2 Recordemos con este ejemplo cuál es la situación:

```
#módulo funciones
def uno():
    print("uno")
    print(f"El nombre de este módulo es {__name__}")
#uno()

#uso_funciones
import funciones
funciones.uno()
```

3 El módulo ___main___

- Las instrucciones ejecutadas en el nivel de llamadas superior del intérprete, ya sea desde un script o interactivamente, se consideran parte del módulo llamado **___main___**, por lo tanto tienen su propio espacio de nombres global.

```
#módulo funciones
print(f"El nombre de este módulo es {__name__}")

if __name__ == "__main__":
    uno()
```

4 Veamos este otro ejemplo:

```
# modulo utiles
def vocales(cadena):
    print(list(filter(lambda l: l.lower() in "aeiou", cadena)))
```

```
# modulo uso_utiles
import utiles

utiles.vocales("Holaaaa!!!!!!")
```

- Primero: ¿qué hace?

4.0.1 ¿Y si queremos invocar el módulo utiles (e invocar a la función vocales) desde la línea de comandos?¿Cómo les pasamos la cadena a analizar?

[ ]:
```
import sys
type(sys.argv)
```

- ¿De qué tipo es argv?
- ¿Qué valores contiene?

[ ]:
```
sys.argv[0]
```

```
# modulo utiles
def vocales(cadena):
    print(list(filter(lambda l: l.lower() in "aeiou", cadena)))

if __name__ == "__main__":
    import sys
    vocales(sys.argv[1])
```

5 Pensemos en las siguientes situaciones

¿Qué estructura usamos si queremos:

- guardar los puntajes cada vez que jugamos a un juego determinado?,
- tener un banco de preguntas para que cada vez que realizamos el repaso de clase pueda acotar por temas?,
- manipular los Python plus de los estudiantes por turnos?

¿Qué tienen todas estas situaciones en común?

###

Necesitamos una estructura que permita que los datos puedan **persistir** cuando la ejecución del programa finalice

6 Algunas consideraciones antes de empezar

- Lo básico: ¿qué es un **archivo**?
- ¿Cómo podemos manipular los archivos desde un programa Python?

7 Manejo de archivos

- Existen funciones predefinidas.

- Si las operaciones fallan, se levanta una **excepción**.
- Los archivos se manejan como objetos que se crean usando la función open.

8 La función open

(https://docs.python.org/3/library/functions.html#open).

Tarea para el hogar. Investigar: ¿qué diferencias hay entre un archivo de texto y uno binario?

9 Veamos este ejemplo

[ ]:
```
archi1 = open('archivo.txt', 'w')
```

- ¿De qué modo se abre este archivo? ¿Qué significa?
- Luego de la instrucción, ¿dónde se encuentra archivo.txt?
- ¿Cuándo puede dar un error esta sentencia?

10 ¿Y este otro ejemplo?

[ ]:
```
archi2 = open('archivo.txt', 'x')
```

- Y en este caso, ¿de qué modo se abre este archivo?
- ¿Cuándo puede dar un error esta sentencia?

11 ¿Y en este caso?

[ ]:
```
archi3 = open('archivo.txt')
```

¿De qué modo se abre este archivo? ¿Da error el código?

- En realidad la función open tiene más argumentos:

```
open(file, mode='r', buffering=-1, encoding=None,
     errors=None, newline=None, closefd=True, opener=None)
```

- **encoding**: sólo para modo texto. Por defecto, la codificación establecida en las configuraciones del sistema

  ```
  archi = open("pp.xxx", "r+", encoding="UTF-8")
  ```

[ ]:
```
import locale
locale.getlocale()
```

11.1 ¿Qué pasa si el archivo no está en la misma carpeta y tenemos que utilizar la ruta completa?

[ ]:
```
import os

ruta = os.path.dirname(os.path.realpath("."))
ruta
```

[ ]:
```
ruta_completa = os.path.join(ruta, "teorias", "ejemplos","clase4", "archivo.
↪txt")
ruta_completa
```

12 ¿Cómo almacenamos datos en un archivo?

- El caso más sencillo: guardanto texto en un archivo.

[ ]:
```
f = open('archivo.txt', 'w')

cad1 = "De los pibes de Malvinas"
cad2 = "Que jamás olvidaré."
f.write("En Argentina nací")
f.write("Tierra del Diego y Lionel")
f.write(cad1)
print(f.write(cad2))
f.close()
```

- **write(cadena):** escribe *cadena* en el archivo y retorna cantidad de caracteres escritos.
- **close():** cierra el archivo.

13 ¿Cómo leemos los datos guardados?

[ ]:
```
f = open('archivo.txt', 'r')
x = f.read(4)
print(f.read())
x
```

- **read(cantidad_bytes):** lee *cantidad_bytes* del archivo.
- Si cantidad_bytes es <0 o no está, lee hasta fin de archivo.
- Retorna "" si EOF.

- **Tarea**: probar el siguiente ejemplo que muestran otras formas de leer caracteres desde un archivo de texto.

[ ]:
```
archi_muchachos = os.path.join(ruta, "teorias", "ejemplos","clase4", "muchachos.
↪txt")
```

[ ]:
```python
def leo_caracteres():
    f = open(archi_muchachos,"r")
    for x in f.read():
        print(x)
    f.close()

def leo_lineas():
    f = open(archi_muchachos,"r")
    print(f.readlines())
    f.close()

def otra_forma():
    f = open(archi_muchachos,"r")
    for linea in f:
        print(linea)
    f.close()
```

[ ]:
```python
def main():
    print('Leo caracteres')
    leo_caracteres()
    print('-' * 20)
    print('Leo lineas')
    leo_lineas()
    print('-' * 20)
    print('Otra forma')
    otra_forma()


if __name__ == "__main__":
    otra_forma()
```

14 ¿Qué pasa si necesito guardar información que tiene una estructura?

- Pensemos en estos ejemplos:
  - Los puntajes cada vez que juego a un juego. Información tipo: nombre jugador, puntaje, fecha.
  - El banco de preguntas: tema, enunciado, respuesta correcta.
  - Los Python plus de los estudiantes por turnos: turno, nombre, apellido, num_legajo, cantidad_puntos, etc.
- En estos casos también podría usar un archivo de texto: ¿cómo se les ocurre?

15 Algunas posibilidades

```
'equipo: BESTIA - esport: CSGO - pais: Argentina'

---
equipo:BESTIA
esport:CSGO
pais:Argentina
---

'BESTIA-CSGO-Argentina'

'BESTIA*CSGO*Argentina*'
```

- ¿Pros y contras?

16 Hay otras formas mejores...

17 JSON (JavaScript Object Notation)

- Es un formato de intercambio de datos muy popular. Por ejemplo:

```
{"equipo": "BESTIA",
 "esport": "CSGO",
 "pais": "Argentina"}
o
[{"equipo": "BESTIA",
  "esport": "CSGO",
  "pais": "Argentina"},
  {"equipo": "9z",
  "esport": "CSGO",
  "pais": "Argentina"}]
```

- +Info
- Veamos este ejemplo: https://developers.mercadolibre.com.ar/es_ar/categorias-y-publicaciones#close

18 Módulo json

- Python tiene un módulo que permite trabajar con este formato.
- Para usarlo, debemos importarlo.

[ ]:
```
import json
```

- Permite serializar objetos.
  - serializamos con: **dumps()** y **dump()**.
  - deserializamos con: **loads()** y **load()**.
- Más info en: https://docs.python.org/3/library/json.html

18.0.1 Veamos este ejemplo

- Generamos un archivo con bandas de distintas ciudades:
  - Tenemos: nombre de la banda, ciudad en la que se generó y una referencia a su trabajo.
  - Empecemos por La Plata...

[ ]:
```
import json

archivo = open("bandas.txt", "w")
datos = [
    {"nombre": "William Campbell", "ciudad": "La Plata", "ref": "www.instagram.
 ↪com/williamcampbellok"},
    {"nombre": "Buendia", "ciudad": "La Plata", "ref":"https://buendia.bandcamp.
 ↪com/"},
    {"nombre": "Lúmine", "ciudad": "La Plata", "ref": "https://www.instagram.
 ↪com/luminelp/"}]
json.dump(datos, archivo)
archivo.close()
```

- ¿De qué tipo es la variable **datos**?

[ ]:
```
# Ahora accedemos a los datos guardados
import json

archivo = open("bandas.txt", "r")
datos = json.load(archivo)
print(datos)
```

- ¿De qué tipo de **datos**?

[ ]:
```
datos_a_mostrar = json.dumps(datos, indent=4)
print(datos_a_mostrar)
archivo.close()
```

- ¿De qué tipo es **datos_a_mostrar**?

19 CSV: ¿más formatos?

- CSV (Comma Separated Values).
- Es un formato muy común para importar/exportar desde/hacia hojas de cálculo y bases de datos.
- Ejemplo:

```
nombre,ciudad,ref
William Campbell,La Plata,www.instagram.com/williamcampbellok
Buendia,La Plata,https://buendia.bandcamp.com/
Lúmine,La Plata,https://www.instagram.com/luminelp/
```

- +Info: https://docs.python.org/3/library/csv.html

- PEP 305

20 Datasets

- Hay muchos datasets disponibles de muchas temáticas:
- Datos de gestión de gobiernos:
  - Har un portal de datos de Argentina, de CABA, de La Plata
  - Datos del Banco mundial
  - Portal Kagle
  - IMDB
  - Y muchos más...
- Muchos son datos abiertos, pero otros... no tanto...
- ¡PRESTAR ATENCIÓN a la licencias y requisitos para su uso!

21 ¿Qué vemos en netflix?

Vamos a trabajar con el archivo: netflix_titles.csv

[ ]:
```
import csv

ruta = os.path.dirname(os.path.realpath("."))
ruta_archivo = os.path.join(ruta, "teorias", "ejemplos","clase4",␣
 ↪"netflix_titles.csv")
ruta_archivo
```

[ ]:
```
archivo = open(ruta_archivo, "r")
csvreader = csv.reader(archivo, delimiter=',')
```

[ ]:
```
#encabezado = csvreader.__next__()
encabezado = next(csvreader)
print(encabezado)
```

[ ]:
```
archivo.close()
```

22 El módulo csv

- Hay que importarlo.
- **csv.reader**: crea un objeto **iterador** que nos permite recorrer las líneas del archivo.
- ¿Por qué incluimos el paŕametro **delimiter**? ¿Dialectos?

```
csvreader = csv.reader(archivo, delimiter=',')
```

23 ¿Qué películas argentinas vemos?

[ ]:
```
archivo = open(ruta_archivo, "r")
csvreader = csv.reader(archivo, delimiter=',')

#encabezado = csvreader.__next__()
encabezado = next(csvreader)
print(encabezado)
```

[ ]:
```
for linea in csvreader:
    if linea[1] == "Movie" and linea[5] == "Argentina":
        print(f"{linea[2]:<40} {linea[3]}")

archivo.close()
```

23.1 ¿De qué tipo es línea?

24 Otra solución ...

[ ]:
```
archivo = open(ruta_archivo, "r")
csvreader = csv.reader(archivo, delimiter=',')

shows_ar = filter(lambda x:  x[5] == "Argentina" and x[1] == "Movie", csvreader)
for elem in shows_ar:
    print(f"{elem[2]:<40} {elem[3]}")

print(shows_ar)
archivo.close()
```

25 Otra forma de acceder: csv.DictReader

[ ]:
```
archivo = open(ruta_archivo, "r")
csvreader = csv.DictReader(archivo, delimiter=',')

shows_ar = filter(lambda x:  x["country"] == "Argentina" and x["type"] ==␣
 ↪"Movie", csvreader)

for linea in shows_ar:
    print(linea["title"])

archivo.close()
```

26 Creamos nuestro archivo csv de bandas de música

- **csv.writer:** retorna un objeto que convierte los datos con los que trabajamos en el programa en cadenas con el formato delimitadas con el separador correspondiente.

[ ]:
```
import csv
import json

archivo = open("bandas.txt")
archivo_csv = open("bandas.csv", "w")

bandas = json.load(archivo)

writer = csv.writer(archivo_csv)
writer.writerow(["Nombre", "Ciudad de procedencia", "Refencias"])
for banda in bandas:
    writer.writerow([banda["nombre"], banda["ciudad"], banda["ref"]])

archivo.close()
archivo_csv.close()
```

27 Lo leemos

[ ]:
```
archivo_cvs = open("bandas.csv", "r")
csvreader = csv.reader(archivo_cvs, delimiter=',')

for linea in csvreader:
    print(linea)

archivo_csv.close()
```

28 La sentencia with

La sentencia with crea un objeto denominado **runtime context** o **contexto de tiempo de ejecución** que permite ejecutar un grupo de sentencias bajo el control de un **administrador de contexto** o **context manager**. ¿Qué es esto?

> Un administrador de contexto permite asignar y liberar recursos cuando se desee.

El ejemplo típico se da en el acceso a arhivos, ya que son recursos externos a nuestros programas que requieren una gestión adecuada.

[ ]:
```
with open("bandas.csv") as archi_bandas:
    csv_reader = csv.reader(archi_bandas, delimiter=',')
    encabezado, data = next(csv_reader), list(csv_reader)

print(encabezado)
```

```
for linea in data:
    print(linea)
```

¿Dónde se cierra el archivo?

29 Desafío

- Dado el conjunto de datos con series y películas de Netflix, queremos:

  1- guardar en otro archivo, en formato json, las peliculas realizadas por más de un país.

  2- los cinco (5) países con más producciones en Netflix, durante el año 2019.

30 Actividad 1 por Python plus

30.0.1 Veamos en qué consiste ...