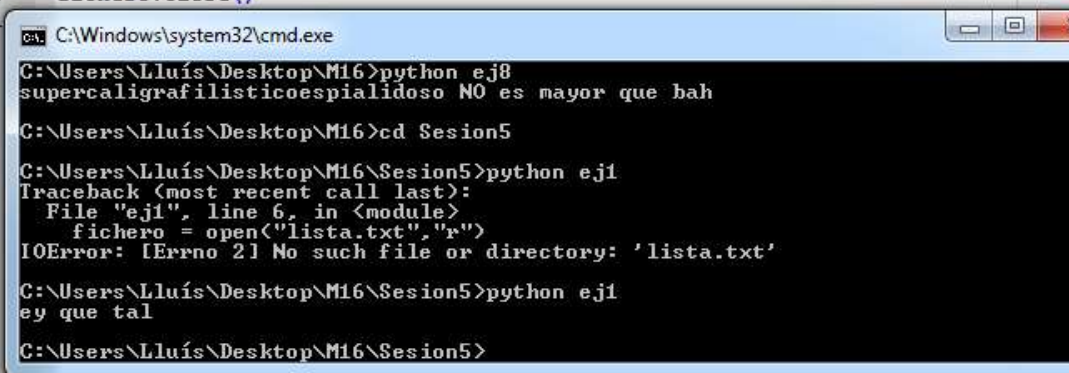


Sessió5:Diversos temes pendants

Ejercicio 1

```
1  #!/usr/bin/python
2  # -*- coding: utf-8 -*-
3
4
5  #Abrimos el fichero en modo lectura
6  fichero = open("lista.txt","r")
7  contenido = fichero.read()
8
9  print contenido
10 fichero.close()
11
```



```
C:\Windows\system32\cmd.exe
C:\Users\Lluís\Desktop\M16>python ej8
supercaligrafilisticoespialidoso NO es mayor que bah

C:\Users\Lluís\Desktop\M16>cd Sesion5

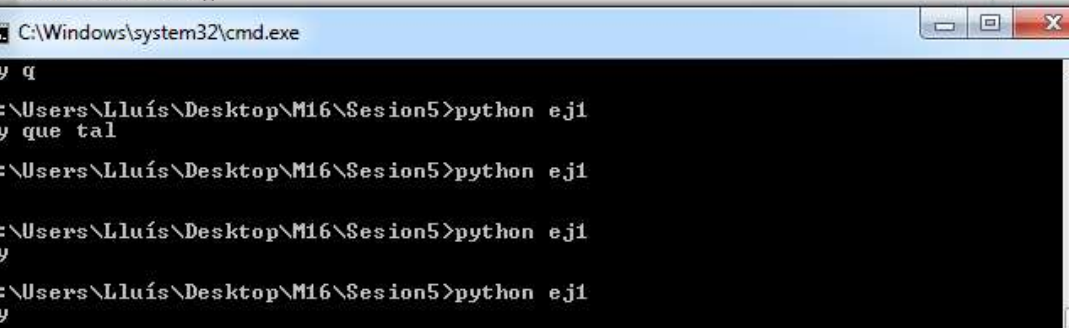
C:\Users\Lluís\Desktop\M16\Sesion5>python ej1
Traceback (most recent call last):
  File "ej1", line 6, in <module>
    fichero = open("lista.txt","r")
IOError: [Errno 2] No such file or directory: 'lista.txt'

C:\Users\Lluís\Desktop\M16\Sesion5>python ej1
ey que tal

C:\Users\Lluís\Desktop\M16\Sesion5>
```

Indicando nsotros el numero de bytes que puede leer:

```
1  #!/usr/bin/python
2  # -*- coding: utf-8 -*-
3
4
5  #Abrimos el fichero en modo lectura
6  fichero = open("lista.txt","r")
7  contenido = fichero.read(3)
8
9  print contenido
10 fichero.close()
11
```



```
C:\Windows\system32\cmd.exe
ey q

C:\Users\Lluís\Desktop\M16\Sesion5>python ej1
ey que tal

C:\Users\Lluís\Desktop\M16\Sesion5>python ej1
e

C:\Users\Lluís\Desktop\M16\Sesion5>python ej1
ey

C:\Users\Lluís\Desktop\M16\Sesion5>python ej1
ey
```

Ejercicio 2

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

#Abrimos el fichero en modo lectura
fichero = open("lista.txt", "r")

print fichero.readline()
print fichero.readline()
print fichero.readline()
fichero.close()
```

```
C:\Windows\system32\cmd.exe
C:\Users\Lluís\Desktop\M16\Sesion5>python ej2
C:\Users\Lluís\Desktop\M16\Sesion5>python ej2
ey que tal
C:\Users\Lluís\Desktop\M16\Sesion5>python ej2
ey que tal
C:\Users\Lluís\Desktop\M16\Sesion5>python ej2
ey que tal
C:\Users\Lluís\Desktop\M16\Sesion5>
```

Como le pedimos que lea dos líneas en las cuales no hay texto, nos imprime dos líneas vacías.

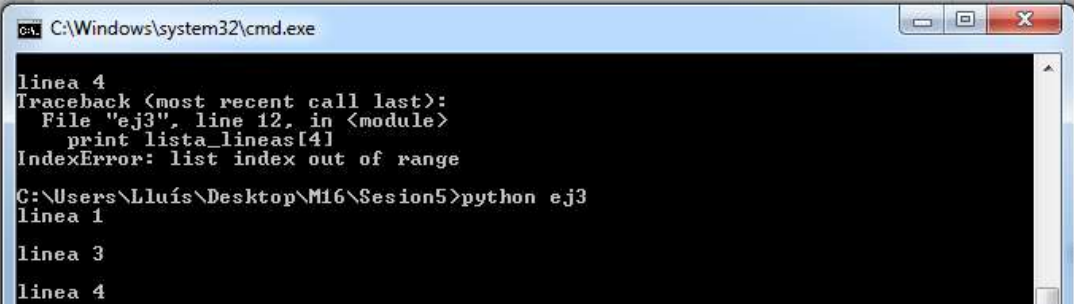
Modificado:

```
1  #!/usr/bin/python
2  # -*- coding: utf-8 -*-
3
4
5  #Abrimos el fichero en modo lectura
6  fichero = open("lista.txt", "r")
7
8  print fichero.readline()
9  while fichero.readline() != "":
10     print fichero.readline()
11  fichero.close()
12
```

```
C:\Windows\system32\cmd.exe
File "ej2", line 8, in <module>
  infile.readline()
ValueError: Mixing iteration and read methods would lose data
C:\Users\Lluís\Desktop\M16\Sesion5>python ej2
ey que tal
estas
C:\Users\Lluís\Desktop\M16\Sesion5>python ej2
ey que tal
estas
C:\Users\Lluís\Desktop\M16\Sesion5>
```

Ejercicio 3

```
1  #!/usr/bin/python
2  # -*- coding: utf-8 -*-
3
4
5  #Abrimos el fichero en modo lectura
6  fichero = open("lista.txt","r")
7
8  lista_lineas = fichero.readlines()
9
10 print lista_lineas[1]
11 print lista_lineas[3]
12 print lista_lineas[4]
13
14 fichero.close()
15
```



```
linea 4
Traceback (most recent call last):
  File "ej3", line 12, in <module>
    print lista_lineas[4]
IndexError: list index out of range

C:\Users\Lluís\Desktop\M16\Sesion5>python ej3
linea 1
linea 3
linea 4
```

Imprime las líneas que le pasamos por parametro

Para que las imprima en orden inverso cambiamos el orden de los parámetros y ya esta

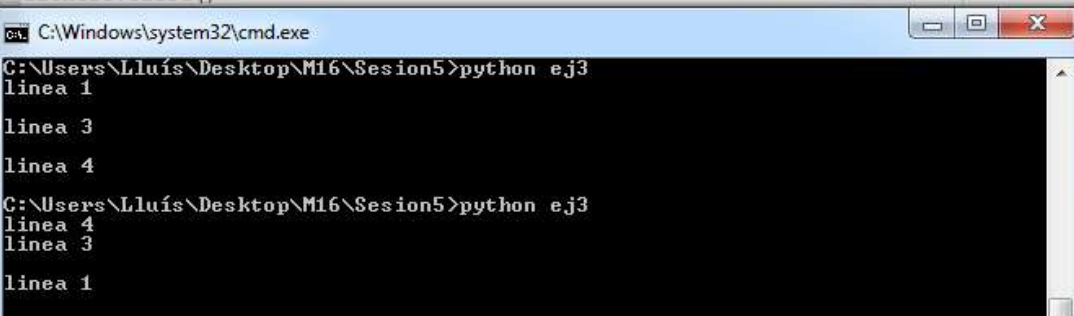
```
#!/usr/bin/python
# -*- coding: utf-8 -*-

#Abrimos el fichero en modo lectura
fichero = open("lista.txt","r")

lista_lineas = fichero.readlines()

print lista_lineas[4]
print lista_lineas[3]
print lista_lineas[1]

fichero.close()
```

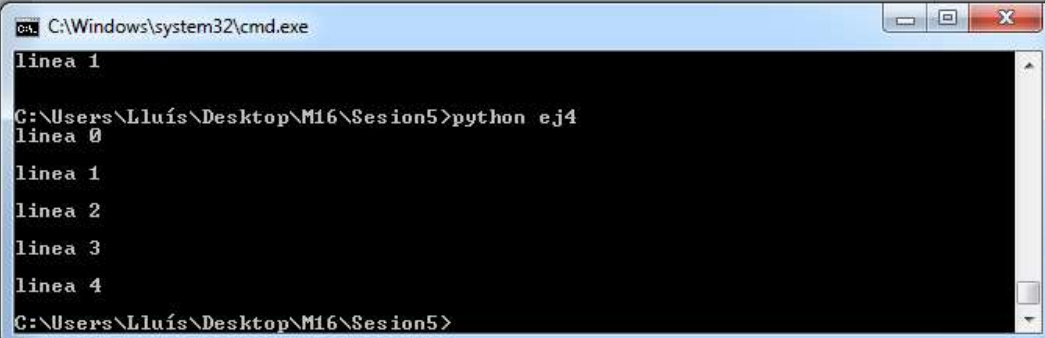


```
C:\Users\Lluís\Desktop\M16\Sesion5>python ej3
linea 1
linea 3
linea 4

C:\Users\Lluís\Desktop\M16\Sesion5>python ej3
linea 4
linea 3
linea 1
```

Ejercicio 4

```
1  #!/usr/bin/python
2  # -*- coding: utf-8 -*-
3
4
5  #Abrimos el fichero en modo lectura
6  fichero = open("lista.txt", "r")
7
8  for linea in fichero:
9      print linea
10
11  fichero.close()
12
```



```
C:\Windows\system32\cmd.exe
linea 1
C:\Users\Lluís\Desktop\M16\Sesion5>python ej4
linea 0
linea 1
linea 2
linea 3
linea 4
C:\Users\Lluís\Desktop\M16\Sesion5>
```

En la variable fichero tenemos cargado el archivo lista.txt y con el for recorremos cada línea y la vamos printando.

Ejercicio 5

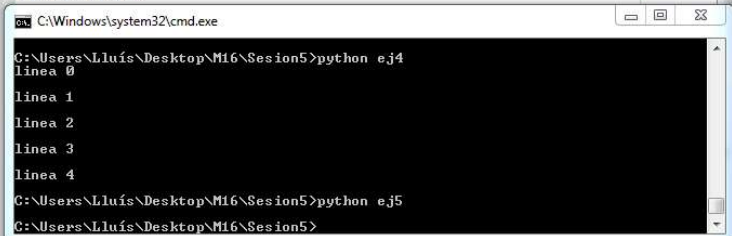
```
#!/usr/bin/python
# -*- coding: utf-8 -*-

un_texto = "Hola mundo"
otro_texto = "Adios mundo"


#abrimos el fichero en modo escritura
#si el archivo ya existe, se borrara el contenido

fichero = open("nuevo.txt", "w")
fichero.write(un_texto)
fichero.write(otro_texto)

fichero.close()
```



```
C:\Windows\system32\cmd.exe
C:\Users\Lluís\Desktop\M16\Sesion5>python ej4
linea 0
linea 1
linea 2
linea 3
linea 4
C:\Users\Lluís\Desktop\M16\Sesion5>python ej5
C:\Users\Lluís\Desktop\M16\Sesion5>
```



```
nuevo.txt: Bloc de notas
Archivo Edición Formato Ver Ayuda
Hola mundoAdios mundo
```

Crea el archivo nuevo.txt y añade las variables un_texto y otro_texto

Ejercicio 6



```
1  #! /usr/bin/python
2  # -*- coding: utf-8 -*-
3
4
5  #abrimos el fichero en modo escritura
6  #si el archivo ya existe, se borrara el contenido
7
8  fichero = open("nuevo.txt", "w")
9
10 for i in range(1,5):
11     texto = "Linea %i\n" % i
12     fichero.write(texto)
13
14 fichero.close()
15
```

```
C:\Windows\system32\cmd.exe
C:\Users\Lluís\Desktop\M16\Sesion5>python ej6
C:\Users\Lluís\Desktop\M16\Sesion5>
```

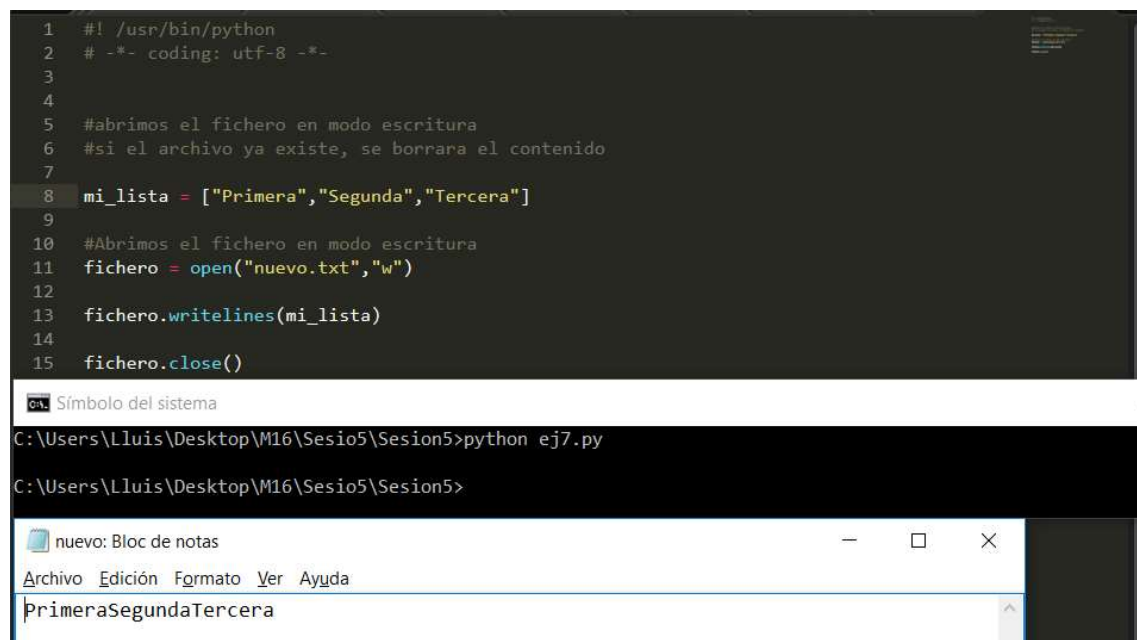
nuevo.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

Linea 1
Linea 2
Linea 3
Linea 4

Escribe o reescribe en el caso de que exista el archivo nuevo.txt, añade las líneas dentro del parámetro que le marcamos en el for.

Ejercicio 7



```
1  #! /usr/bin/python
2  # -*- coding: utf-8 -*-
3
4
5  #abrimos el fichero en modo escritura
6  #si el archivo ya existe, se borrara el contenido
7
8  mi_lista = ["Primera", "Segunda", "Tercera"]
9
10 #Abrimos el fichero en modo escritura
11 fichero = open("nuevo.txt", "w")
12
13 fichero.writelines(mi_lista)
14
15 fichero.close()
16
```

```
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python ej7.py
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>
```

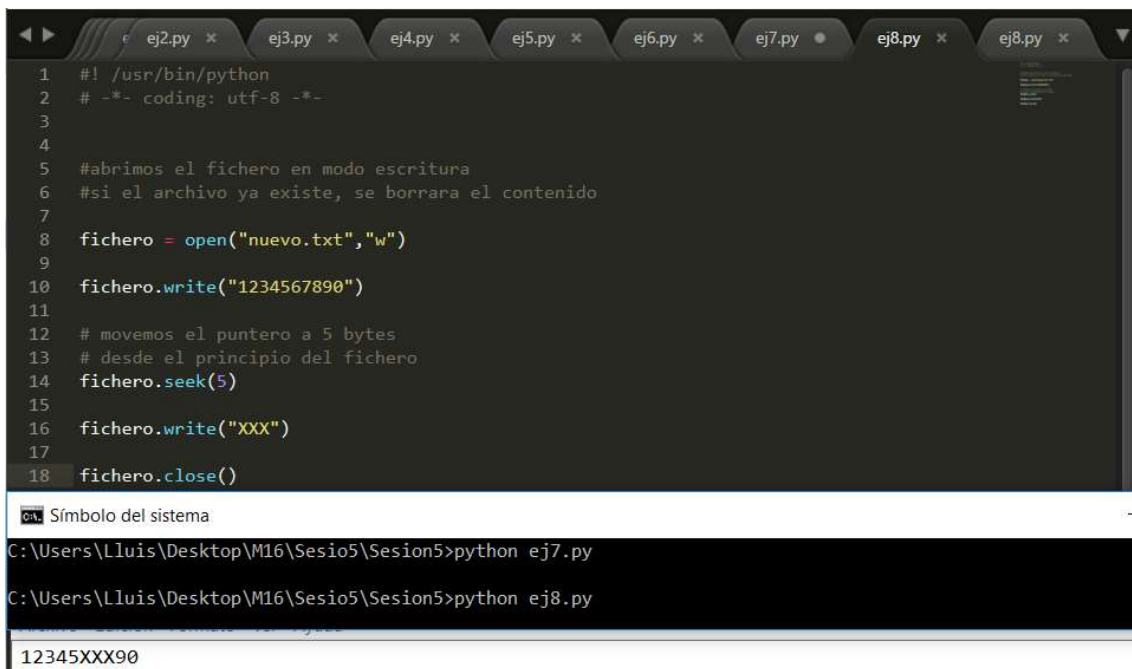
nuevo: Bloc de notas

Archivo Edición Formato Ver Ayuda

PrimeraSegundaTercera

Nos escribe la lista en el archivo nuevo.txt

Ejercicio 8



The screenshot shows a code editor with a Python script and a terminal window. The script, named `ej8.py`, opens a file `nuevo.txt` in write mode, writes the string `"1234567890"`, moves the file pointer to the 5th byte using `seek(5)`, overwrites the next three characters with `"XXX"`, and then closes the file. The terminal shows the command `python ej8.py` being executed, resulting in the output `12345XXX90`.

```
1 #! /usr/bin/python
2 # -*- coding: utf-8 -*-
3
4
5 #abrimos el fichero en modo escritura
6 #si el archivo ya existe, se borrara el contenido
7
8 fichero = open("nuevo.txt","w")
9
10 fichero.write("1234567890")
11
12 # movemos el puntero a 5 bytes
13 # desde el principio del fichero
14 fichero.seek(5)
15
16 fichero.write("XXX")
17
18 fichero.close()
```

Símbolo del sistema

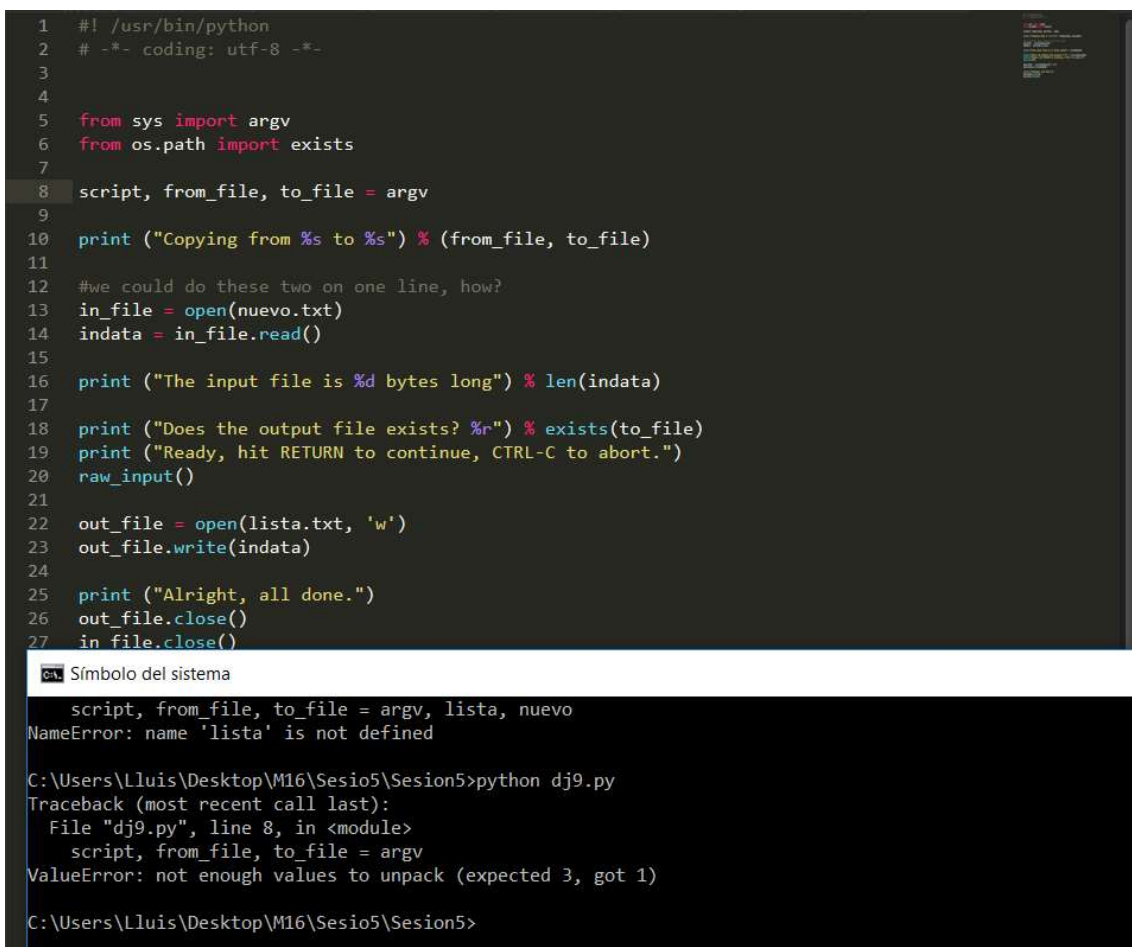
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python ej7.py

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python ej8.py

12345XXX90

Escribimos del 1234567890, el `fichero.seek` sitúa el puntero en el byte 5 y sobrescribe 3 X.

Ejercicio 9



The screenshot shows a code editor with a Python script and a terminal window. The script, named `dj9.py`, imports `argv` from `sys` and `exists` from `os.path`. It takes three arguments: `script`, `from_file`, and `to_file`. It prints a message about copying, checks if the output file exists, and prompts the user to hit RETURN. It then opens `nuevo.txt`, reads its content, and writes it to `lista.txt`. The terminal shows the command `python dj9.py` being executed, resulting in a `NameError: name 'lista' is not defined` and a `ValueError: not enough values to unpack (expected 3, got 1)`.

```
1 #! /usr/bin/python
2 # -*- coding: utf-8 -*-
3
4
5 from sys import argv
6 from os.path import exists
7
8 script, from_file, to_file = argv
9
10 print ("Copying from %s to %s" % (from_file, to_file))
11
12 #we could do these two on one line, how?
13 in_file = open(nuevo.txt)
14 indata = in_file.read()
15
16 print ("The input file is %d bytes long" % len(indata))
17
18 print ("Does the output file exists? %r" % exists(to_file))
19 print ("Ready, hit RETURN to continue, CTRL-C to abort.")
20 raw_input()
21
22 out_file = open(lista.txt, 'w')
23 out_file.write(indata)
24
25 print ("Alright, all done.")
26 out_file.close()
27 in_file.close()
```

Símbolo del sistema

script, from_file, to_file = argv, lista, nuevo

NameError: name 'lista' is not defined

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python dj9.py

Traceback (most recent call last):

File "dj9.py", line 8, in <module>

script, from_file, to_file = argv

ValueError: not enough values to unpack (expected 3, got 1)

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>


El programa tiene un script para copiar todo lo de un archivo a otro. Te dira la longitud en bytes del archivo que vas a copiar, y mirara si existe el archivo en el que vas a copiarlo, si le das a return continua y si le das a control c se para el programa. En caso de darle a return te copia a continuación de lo que tenias la información en el archivo que has puesto que se copie. Y printa alright, all done al acabar.

Para sobrescribir la información seria `out_file = open(to_file, 'w+')`. Para copiar la información al final del archivo seria con `f.seek` y colocándote al final.

Los parámetros que se pasan son el archivo que vas a copiar y al cual lo vas a copiar.

El error que me sale he estado investigando y no consigo saber porque es, he probado en Python V.2 (utilizo la 3) y tampoco. He probado de hacer modificaciones pero si no me da error en la línea 4 me lo da en la 6.(del archivo que esta en el pdf).

Pruebas 2. Matrius en Python.

 Símbolo del sistema - python

```
Type "help", "copyright", "credits" or "license" for more information.
>>> nested = ["hello", 2.0, 5, [10,20]]
>>> elem = nested[3]
>>> elem[0]
10
>>> elem[1]
20
>>> nested[3][1]
20
>>>
```

```
>>> matrix = [[1,2,3],[4,5,6],[7,8,9]]
>>> matrix[1]
[4, 5, 6]
>>> matrix[1][1]
5
>>>
```

Activitat:

```
def producte_matriu_escalar(mat,k):
    for m in range (len(mat)):
        print()
        for n in range (len(mat[m])):
            print((mat[m][n]*k),end=" ")

lista=[[1,2,3],[4,5,6],[7,8,9]]
producte_matriu_escalar(lista,2)
```

 Símbolo del sistema

```
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python matriu2.py

2 4 6
8 10 12
14 16 18
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>
```

Se ha creado una función en la cual le pasamos mat(que es una matriz) y una variable que será x el numero que lo vamos a multiplicar. Hacemos un for para ir recorriendo filas y columnas y multiplicar por el numero que le hemos pasado.

Ejercicios mes sobre funciones

```
1 def saluda():
2     return "hola", "mundo"
3
4 resultat1, resultat2 = saluda()
5 print (resultat1)
6 print (resultat2)
```

C:\> Seleccionar Símbolo del sistema

```
resultat1, resultat2 = saluda()
^
SyntaxError: invalid syntax
>>>
^C
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python retornmulti.py
hola
mundo

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>
```

```
1 def saluda():
2     return "hola", "mundo"
3
4 hola = saluda()
5 print (hola)
```

C:\> Símbolo del sistema

```
>>>
^C
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python retornmulti.py
hola
mundo

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python retornmulti.py
('hola', 'mundo')

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>
```

```
1 def f(a, b, c='A', d='B'):
2     return list(str(a) + str(b) + str(c) + str(d))
3
4 print (f(1,2))
5 print (f(1,2,3))
6 print (f(1,2,3,4))
```

C:\> Símbolo del sistema

```
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python retornmulti.py
('hola', 'mundo')

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python retornmulti.py
['1', '2', 'A', 'B']
['1', '2', '3', 'B']
['1', '2', '3', '4']

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>
```


Por defecto tenemos la función B, en la cual las 2 primeras letras no tienen ningún valor y las dos ultimas tenemos asignado 'A' y 'B'.

En el primer print añadimos 1 y 2 a las variables a y b, c y d se printan con el valor que ya esta pasado al definir la función.

En el segundo print lo mismo y machacamos el valor de C, y printamos el nuevo.

En el tercer print printamos los valores(también c y d una vez son machacados). En el caso de llamar a la función SIEMPRE tendremos que poner dos valores, si no no funcionara porque a y b no tendrna valor.

Poniendo valores por defecto:

```
1 def f(a='1', b='2', c='A', d='B'):  
2     return list(str(a) + str(b) + str(c) + str(d))  
3  
4 print (f(1,2))  
5 print (f(1,2,3))  
6 print (f(1,2,3,4))
```

CA. Símbolo del sistema

```
['1', '2', 'A', 'B']  
['1', '2', '3', 'B']  
['1', '2', '3', '4']  
  
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python retornmulti.py  
['1', '2', 'A', 'B']  
['1', '2', '3', 'B']  
['1', '2', '3', '4']  
  
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>
```

Igualmente va a imprimir lo que le pasemos al llamar a la función. (comprobado cambiando por ejemplo en el print un dos por 3. Aunque b en la función sea igual a 2, si al llamarla le ponemos 3 sera 3. Esto es por el return.

```
def f(a='1', b='2', c='A', d='B'):  
    return list(str(a) + str(b) + str(c) + str(d))  
  
print (f(1,3))  
print (f(1,2,3))  
print (f(1,2,3,4))
```

CA. Símbolo del sistema

```
'1', '2', 'A', 'B']  
'1', '2', '3', 'B']  
'1', '2', '3', '4']  
  
:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python retornmulti.py  
'1', '3', 'A', 'B']  
'1', '2', '3', 'B']  
'1', '2', '3', '4']  
  
:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>
```

Más pruebas de ejercicios:

```
def imprimeix_llista(nom_llista, *coses):
    print ("\nLlista de: ", nom_llista)

    for cosa in coses:
        print (cosa)

imprimeix_llista("Articles de ferreteria", "tenalles", "martell", "tornavis", "bronca")
imprimeix_llista("Cicles de grau superior d'informatica", "ASIX", "DAM", "DAW")
```

Símbolo del sistema

```
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python 33funcions2.py

Llista de: Articles de ferreteria
tenalles
martell
tornavis
bronca

Llista de: Cicles de grau superior d'informatica
ASIX
DAM
DAW

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>
```

Definimos una función de es imprimeix lista, donde apasamos el nom llista y un argumento(lista de cosas).

Printamos llista de y el nombre de la lista, y con un for imprimimos.

Despues llamamos a las funciones y les pasamos el nombre y la lista de cosas para que se ejecute esta y printe algo.

```
1 def imprimir_dades(nom, **dades):
2     print("\nDades de ", nom)
3
4     for clau in dades:
5         print (clau + ": " + dades[clau])
6
7     imprimir_dades("Antoni", edat= "20", matriculat = "si", beca = "no")
8     imprimir_dades("Joan", edat= "22", matriculat = "si", beca = "si", import_beca = "550")
```

Seleccionar Símbolo del sistema

```
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python funciones33.py

Dades de Antoni
beca: no
edat: 20
matriculat: si

Dades de Joan
beca: si
edat: 22
import_beca: 550
matriculat: si

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>
```

Tenemos una función imprimir_dades en la que le pasamos un nombre y un diccionario, con palabras y un valor.

Al llamar a la función imprime dades de y el nombre que la hayamos pasado, después hay un for para recorrer las palabras de **dades, imprimiendo la palabra del diccionario y el “significado”/valor.

Mas ejercicios de practica:

```

1 v_global = "Soc global"
2
3 def funcio():
4     v_local = "Soc local"
5     print ("La variable global diu:", v_global)
6     print ("La variable local diu:", v_local)
7
8 funcio()
9
10 print (v_global)
11
12 print (v_local)

```

```

C:\ Símbolo del sistema
File "34ej1.py", line 10
    print v_global
        ^
SyntaxError: Missing parentheses in call to 'print'

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python 34ej1.py
La variable global diu: Soc global
La variable local diu: Soc local
Soc global
Traceback (most recent call last):
  File "34ej1.py", line 12, in <module>
    print (v_local)
NameError: name 'v_local' is not defined

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>

```

La variable `v_local` no se puede imprimir, porque no está declarada fuera de la función. La global sí porque sí que está declarada.

```

1 la_variable = "Soc global"
2
3 def funcio():
4     la_variable = "Soc local"
5     print ("La variable diu:", la_variable)
6
7 funcio()
8
9 print (la_variable)

```

```

C:\ Símbolo del sistema
    print (v_local)
NameError: name 'v_local' is not defined

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python 34ej1.py
La variable diu: Soc local
Traceback (most recent call last):
  File "34ej1.py", line 9, in <module>
    print (la_variable)
NameError: name 'la_variable' is not defined

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python 34ej1.py
La variable diu: Soc local
Soc global

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>

```

Este ejercicio nos hace ver, que podemos tener una variable con un valor asignado dentro de una función, y con otro valor fuera de la función. Nos imprime el valor de la función y el de fuera.

```
1 variable_global = "Soc global"
2
3 def funcio ():
4     global variable_global
5     variable_global="Ja no soc global"
6     print("la variable variable_global diu", variable_global)
7
8     global nova_global
9     nova_global = "No desapareix despres d'executar la funcio"
10    print(nova_global)
11
12
13 funcio()
14 variable_global
15 nova_global
```

Símbolo del sistema

```
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python 34ej2.py
la variable variable_global diu Ja no soc global
No desapareix despres d'executar la funcio
```

El ejercicio nos permite ver que podemos tener dos variables globales. Se ejecuta la función y nos imprime las dos variables globales declaradas dentro.

Ejercicios sobre excepciones

```
1 dividend = 1
2 divisor = 0
3
4 resultat = dividend/divisor
5
6 print ("El resultat de la divisio es: ", resultat)
```

Símbolo del sistema

```
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python excepcion.py
Traceback (most recent call last):
  File "excepcion.py", line 4, in <module>
    resultat = dividend/divisor
ZeroDivisionError: division by zero
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python excepcion.py
```

Da error porque hace la división entre 0, necesitaríamos hacer una excepción.

```
1 dividend = 1
2 divisor = 0
3
4 try:
5     resultat = dividend/divisor
6     print ("El resultat de la divisio es: ", resultat)
7
8 except:
9     if divisor == 0:
10        print ("No es pot dividir por zero")
```

Símbolo del sistema

```
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python 34ej2.py
la variable variable_global diu Ja no soc global
No desapareix despres d'executar la funcio

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python excepcion.py
No es pot dividir por zero
```

Ahora si se ejecuta bien el programa, porque con el try y el except, añadimos una excepción, en este

```
1 dividend = "7"
2 divisor = 0
3
4 try:
5     resultat = dividend/divisor
6     print ("El resultat de la divisio es: ", resultat)
7
8 except ZeroDivisionError:
9     print ("No es pot dividir por zero")
10
11 except TypeError:
12     print ("Tipus de dades incorrectes")
```

Selecció de Símbol del sistema

```
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python excepcio.py
No es pot dividir por zero

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python excepcio.py
Tipus de dades incorrectes

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>
```

En un mismo script podemos tener varias excepciones, en este caso imprime que el tipo de dato es incorrecto porque he puesto la variable entre “”.

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5\excepcion.py - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

ej8.py x dj9.py x 34ej1.py x 34ej2.py x excepcio.py

```
1 dividend = 7
2 divisor = 2
3
4 try:
5     resultat = dividend/divisor
6
7 except ZeroDivisionError:
8     print ("No es pot dividir por zero")
9
10 except TypeError:
11     print ("Tipus de dades incorrectes")
12
13 else:
14     print ("El resultat de la divisio es: ", resultat)
```

Selecció de Símbol del sistema

```
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python excepcio.py
Tipus de dades incorrectes

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python excepcio.py
El resultat de la divisio es: 3.5

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>
```

En la prueba del PDF sigue estando mal, ya que esta el 7 con “” y pone tipo de datos incorrectos. Para que funcione hay que dejar el 7 y entonces se ejecuta el else.



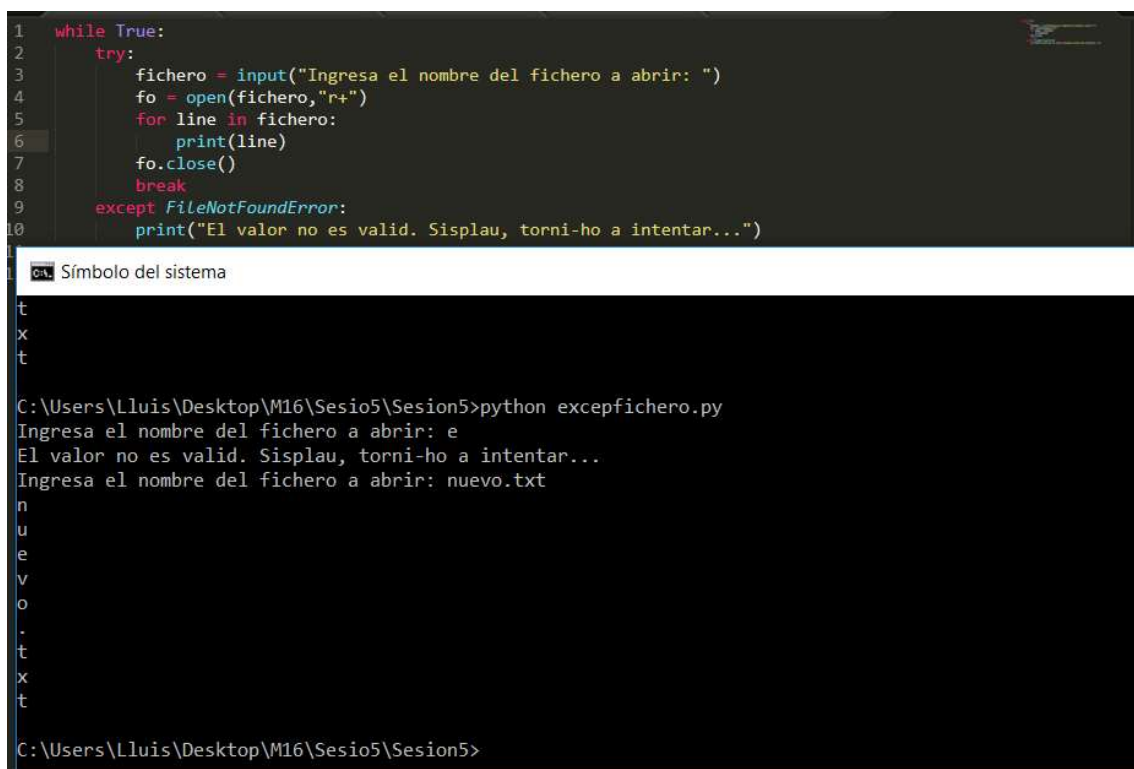
```
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5\excepcion.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

1 while True:
2     try:
3         n= input("Sisplau escrigui un enter: ")
4         n= int(n)
5         break
6     except ValueError:
7         print("El valor no es valid. Sisplau, torni-ho a intentar...")
8 print("Molt be, el valor es valid!")

Símbolo del sistema

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python excepcion.py
Sisplau escrigui un enter: 4.5
El valor no es valid. Sisplau, torni-ho a intentar...
Sisplau escrigui un enter: 4
Molt be, el valor es valid!
```

El archivo se va a ejecutar siempre, nos pedirá un numero y en caso de no ser int saltara la excepción, en caso de que sea valido saldrá el ultimo print.



```
1 while True:
2     try:
3         fichero = input("Ingresa el nombre del fichero a abrir: ")
4         fo = open(fichero,"r+")
5         for line in fichero:
6             print(line)
7         fo.close()
8         break
9     except FileNotFoundError:
10        print("El valor no es valid. Sisplau, torni-ho a intentar...")

Símbolo del sistema

t
x
t

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python excepfichero.py
Ingresa el nombre del fichero a abrir: e
El valor no es valid. Sisplau, torni-ho a intentar...
Ingresa el nombre del fichero a abrir: nuevo.txt
n
u
e
v
o
.
t
x
t

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>
```

Este es el ejercicio que nos pide realizar, pedimos al usuario el nombre del archivo, lo intentamos abrir y en caso de que el nombre coincida nos printa su contenido y si no salta la excepción y nos pide otro “valor”/nombre.

Ejercicios modulos y paquetes


```

1 variable = "Aquesta variable es al modulo"
2
3 def multiplica(num1, num2):
4     return num1 * num2
5
6 def suma(num1, num2):
7     return num1 + num2
8

```

```

1 import mates
2
3 variable = "Aquesta variable es a l'script principal"
4 print (mates.variable)
5 print (variable)
6
7 resultado = mates.suma(3,2)
8 print (resultado)
9
10 resultado = mates.multiplica(3,2)
11 print (resultado)

```

☞ Símbolo del sistema

```

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python mates2.py
Aquesta variable es al modulo
Aquesta variable es a l'script principal
5
6
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python mates2.py

```

Primero imprime la variable variable del modulo mates, después la del script principal, después hace la llamada a una función del modulo mates y printa el resultado, y vuelve a hacer lo mismo con otra función.

Ejercicios Espais de noms

☞ Seleccionar Símbolo del sistema - python

```

>>> import sys
>>> sys.path
['', 'C:\\Users\\Lluís\\AppData\\Local\\Programs\\Python\\Python35-32\\python35.zip', 'C:\\Users\\Lluís\\AppData\\Local\\Programs\\Python\\Python35-32\\DLLs', 'C:\\Users\\Lluís\\AppData\\Local\\Programs\\Python\\Python35-32\\Lib\\site-packages']
>>>

```

Archivo primer.py

```

1 variable = "Soc al modul 'primer'"
2
3 def f_exemple():
4     print (variable)

```

Archivo segon.py

```

1 variable = "Soc al modul 'segon'"
2
3 def f_exemple():
4     print (variable)

```

Ejercicio que llama a primer.py y segon.py

```

import primer, segon

variable = "Soc a l'script principal"

def f_exemple():
    print (variable)

print ("Cridem a la funcio d'aquest modul:")
f_exemple()

print ('Cridem a la funcio del modul "primer":')
primer.f_exemple()

print ('Cridem a la funcio del modul "segon":')
segon.f_exemple()

print ("Tambe podem accedir a les variables:")
print (variable)
print (primer.variable)
print (segon.variable)

```

Resultado:

```

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python ejimports.py
Cridem a la funcio d'aquest modul:
Soc a l'script principal
Cridem a la funcio del modul "primer":
Soc al modul 'primer'
Cridem a la funcio del modul "segon":
Soc al modul 'segon'
Tambe podem accedir a les variables:
Soc a l'script principal
Soc al modul 'primer'
Soc al modul 'segon'

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>

```

Archivo operaciones:

```

1  numero_pi = 3.14159
2
3  def multiplica(un, dos):
4      return un*dos
5
6  def suma(un, dos):
7      return un+dos
8

```

Archivo que llama a operaciones

```

1  from operaciones import multiplica, suma
2
3  print (suma(21,10))
4  print (multiplica(21,10))

```

C:\ Símbolo del sistema
 Soc a l'script principal
 Cridem a la funcio del modul "primer":
 Soc al modul 'primer'
 Cridem a la funcio del modul "segon":
 Soc al modul 'segon'
 Tambe podem accedir a les variables:
 Soc a l'script principal
 Soc al modul 'primer'
 Soc al modul 'segon'

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python operacions2.py
 31
 210

Imagen del error al añadir una función no declarada

```

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5>python operacions2.py
31
210
Traceback (most recent call last):
  File "operacions2.py", line 5, in <module>
    print (numero_pi + operaciones.numero_pi)
NameError: name 'numero_pi' is not defined

```

Ejercicio paquets:

Tenemos el paquete matematicas y dentro los modulos.

Saco foto a la estructura del directorio.

```

Directorio de C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5\matematiques

30/11/2016  13:05    <DIR>        .
30/11/2016  13:05    <DIR>        ..
30/11/2016  09:59             43 constants.py
30/11/2016  13:57            247 fichero.py
30/11/2016  09:59            124 operacions.py
30/11/2016  13:05              0 __init__.py
              4 archivos            414 bytes
              2 dirs  85.506.150.400 bytes libres

C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5\matematiques>

```

El contenido de los archivos es:

Constants.py:

```
constants.py
1 pi = 3.14159
2 e = 2.71828
3 zeta = 1.64493
4
```

Operacions.py:

```
constants.py fichero.py operacions.py __init__.py
1 def multiplica(uno,dos):
2     return uno * dos
3
4 def suma(uno,dos):
5     return uno + dos
6
7 def quadrat(num):
8     return num * num
```

El __init__.py vacío:

```
constants.py fichero.py operacions.py __init__.py
1 |
```

Fichero.py:

```
constants.py fichero.py operacions.py __init__.py
1 import matematiques.operacions
2 import matematiques.constants
3 radi = 14
4
5 radi_quadrat = matematiques.operacions.quadrat(radi)
6 circumferencia = matematiques.operacions.multiplica(matematiques.constants.pi,radi_quadrat)
7
8 print (circumferencia)
```

El error que me sale y que te he comentado en clase es:

```
C:\Users\Lluís\Desktop\M16\Sesio5\Sesion5\matematiques>python fichero.py
Traceback (most recent call last):
  File "fichero.py", line 1, in <module>
    import matematiques.operacions
ImportError: No module named 'matematiques'
```