



Programación Python

José Javier Galán Hernández :

Programación Python

13. MODULOS

13. MODULOS

Organizar mejor el código teniendo en un solo archivo todas las funciones relacionadas con un tema determinado.

Las usaremos en nuestros scripts cuando sea necesario.

Un modulo es un trozo de código en un archivo que puede ser importado en nuestros programas para ser usado.

Cualquier **archivo .py** es un modulo y puede ser importado usando **import**.

Para importar un modulo se utiliza el nombre del archivo sin la extensión “.py”

Para usar las variables y funciones de un modulo desde un programa deberá usarse su **nombre precedido del nombre del modulo y un punto**

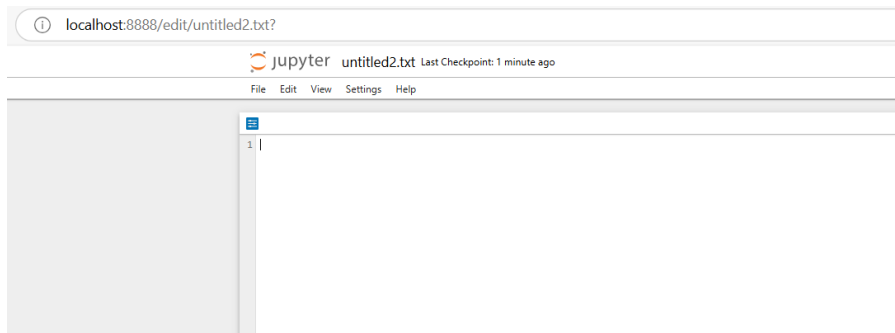
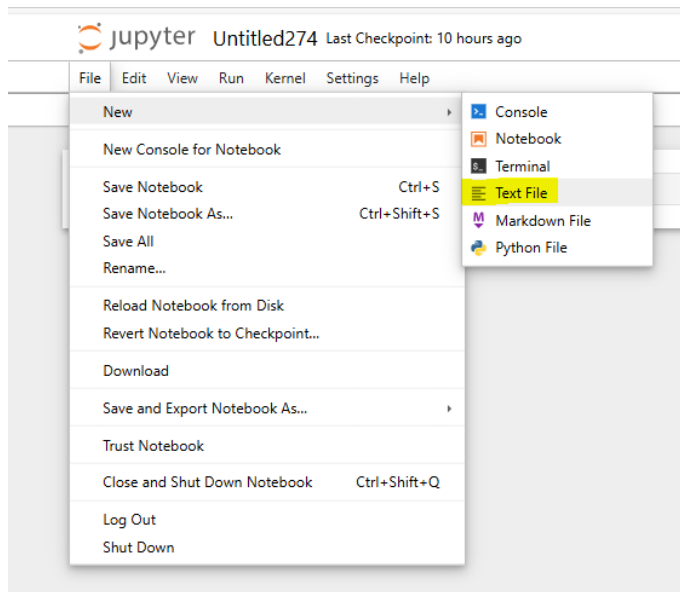
Por defecto los archivos de módulos se buscan el carpeta del usuario, por ejemplo “C:\Users\JOSE”



13. MODULOS. Crear modulo.

En un fichero de texto plano escribimos el código que será importado.

Primero, habremos probado este código en un notebook Python.

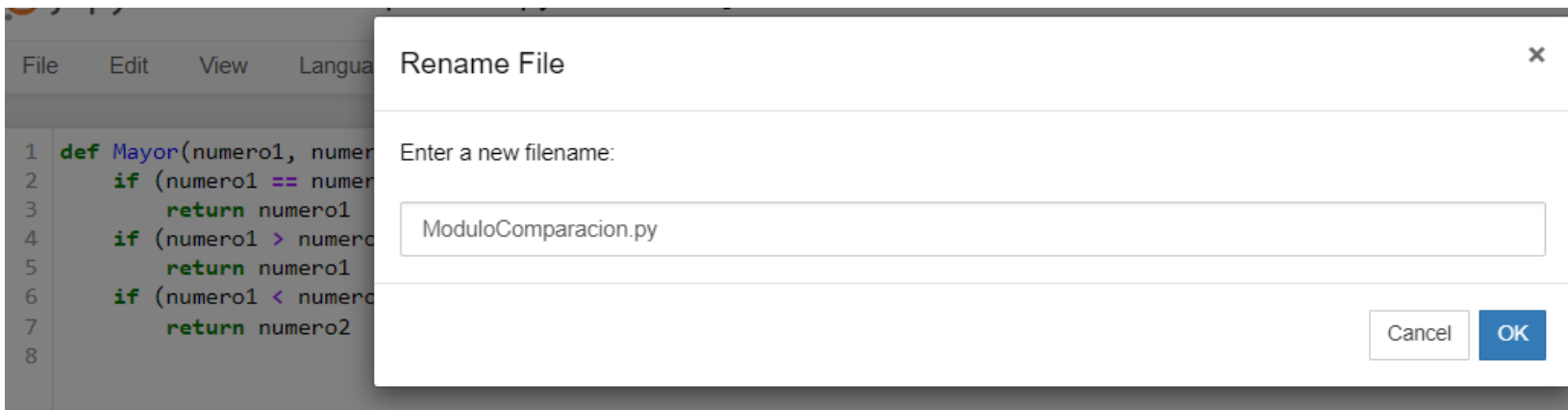


13. MODULOS. Crear modulo.

Copiamos una función que dados 2 números devuelve el mayor. La función se llama “Mayor” y se guarda en un fichero de texto plano llamado “ModuloComparacion.py”

Es fundamental que su extensión sea **.py**

```
def Mayor(numero1, numero2):  
    if (numero1 == numero2):  
        return numero1  
    if (numero1 > numero2):  
        return numero1  
    if (numero1 < numero2):  
        return numero2
```



13. MODULOS. Importar modulo.

```
import ModuloComparacion
```

```
Valor1=1  
Valor2=5  
ModuloComparacion.Mayor(2,5)
```

5


```
import ModuloComparacion as M  
Valor1=1  
Valor2=5  
M.Mayor(2,5)
```

5

Con import podemos importar cualquier modulo.

Después del nombre del módulo importado escribimos un punto (.) para hacer referencia a sus funciones. (Si después del “.” pulsamos la tecla tabulador aparecen las funciones disponibles)

Con “as” podemos poner un alias al módulo importado.



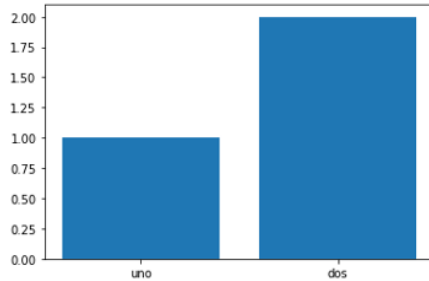
¿Aportan algo
las variables
Valor1 y
Valor2?

13. MODULOS. Importar modulo externo

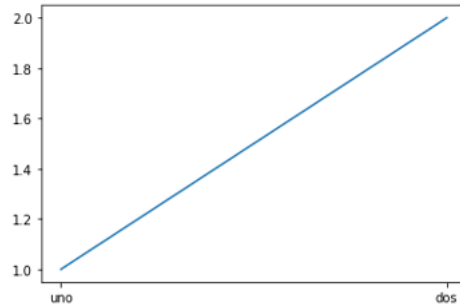
Ejemplo: Librería matplotlib.

%matplotlib inline permite ejecutarlo en Jupyter. (En la última versión de Anaconda no es necesario)

```
%matplotlib inline
import matplotlib.pyplot as grafico
x=['uno','dos']
y=[1,2]
grafico.bar(x,y)
grafico.show()
```



```
%matplotlib inline
import matplotlib.pyplot as grafico
x=['uno','dos']
y=[1,2]
grafico.plot(x,y)
grafico.show()
```



13. MODULOS. Instalar modulo con PIP

pip es un sistema de gestión de paquetes utilizado para instalar y administrar paquetes de software escritos en Python.

Ejemplo: Instalamos OPENPYXL con PIP en Anaconda Prompt

Anaconda Prompt (anaconda3)

Anaconda Prompt (anaconda3)

```
(base) C:\Users\JOSE>pip install openpyxl
Requirement already satisfied: openpyxl in c:\users\jose\anaconda3\lib\site-packages (3.0.9)
Requirement already satisfied: et-xmlfile in c:\users\jose\anaconda3\lib\site-packages (from openpyxl) (1.1.0)

(base) C:\Users\JOSE>
```

Otros entornos sin Anaconda, instalar Python(ver anexo al final)

```
Administrador: Símbolo del sistema
C:\WINDOWS\system32>pip install openpyxl
Collecting openpyxl
  Downloading openpyxl-3.0.3.tar.gz (172 kB)
    | 172 kB 544 kB/s
Collecting jdcals
  Downloading jdcals-1.4.1-py2.py3-none-any.whl (9.5 kB)
Collecting et_xmlfile
  Downloading et_xmlfile-1.0.1.tar.gz (8.4 kB)
Installing collected packages: jdcals, et-xmlfile, openpyxl
  Running setup.py install for et-xmlfile ... done
  Running setup.py install for openpyxl ... done
Successfully installed et-xmlfile-1.0.1 jdcals-1.4.1 openpyxl-3.0.3
C:\WINDOWS\system32>
```




Programación Python

14. Ficheros

14. FICHEROS. OPENPYXL

Importamos Openpyxl, abrimos fichero y mostramos el nombre de sus hojas.

```
import openpyxl

excel_document = openpyxl.load_workbook('C:\\EjerciciosPython\\Modulos\\EjemploExcel.xlsx')
print(excel_document.sheetnames)

['Numeros', 'Letras']
```

Seleccionamos la hoja que deseamos y mostramos el contenido de la celda deseada.

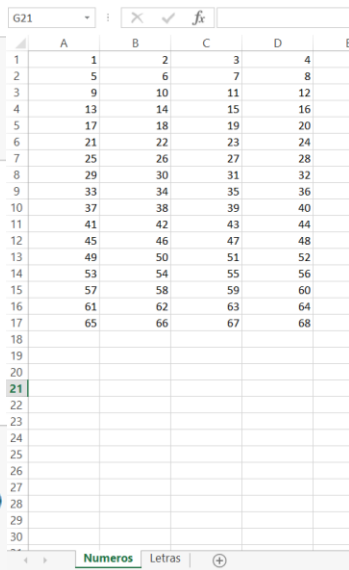
```
import openpyxl

excel_document = openpyxl.load_workbook('C:\\EjerciciosPython\\Modulos\\EjemploExcel.xlsx')
print(excel_document.sheetnames)

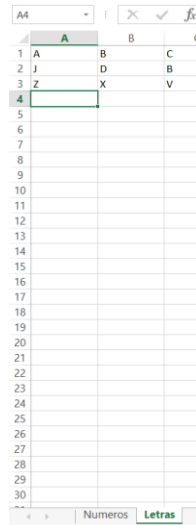
Hoja = excel_document['Numeros'] # Selecciono la hoja de excel que deseo
print(Hoja['A2'].value) # Muestro la casilla indicada
```

```
['Numeros', 'Letras']
```

```
5
```



	A	B	C	D	E
1	1	2	3	4	
2	5	6	7	8	
3	9	10	11	12	
4	13	14	15	16	
5	17	18	19	20	
6	21	22	23	24	
7	25	26	27	28	
8	29	30	31	32	
9	33	34	35	36	
10	37	38	39	40	
11	41	42	43	44	
12	45	46	47	48	
13	49	50	51	52	
14	53	54	55	56	
15	57	58	59	60	
16	61	62	63	64	
17	65	66	67	68	
18					
19					
20					
21					
22					
23					
24					
25					
26					
27					
28					
29					
30					



	A	B	C
1	A	B	C
2	J	D	B
3	Z	X	V
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			
21			
22			
23			
24			
25			
26			
27			
28			
29			
30			



14. FICHEROS . OPENPYXL

Modificamos valores y guardamos.

```
import openpyxl

excel_document = openpyxl.load_workbook('C:\\EjerciciosPython\\Modulos\\EjemploExcel.xlsx')
print(excel_document.sheetnames)

Hoja = excel_document['Numeros']#Selecciono la hoja de excel que deseo
print (Hoja['A2'].value)#Muestro la casilla indicada

Hoja['A2'].value=1984#Modifico su valor
print(Hoja.cell(row = 2, column = 1))#Muestro la casilla
print(Hoja.cell(row = 2, column = 1).value)#Muestro el contenido de la casilla

excel_document.save('C:\\EjerciciosPython\\Modulos\\EjemploExcel.xlsx')#Solo en este momento se modifica el fichero
```

['Numeros', 'Letras']
5
<Cell 'Numeros'.A2>
1984

14. FICHEROS . OPENPYXL

Mostramos un rango concreto.

```
import openpyxl

excel_document = openpyxl.load_workbook('C:\\EjerciciosPython\\Modulos\\EjemploExcel.xlsx')

Hoja = excel_document['Letras']#Selecciono la hoja de excel que deseo

multiple_cells = Hoja['A1':'C3']#rango que quiero mostrar
for row in multiple_cells:#recorro filas
    for cell in row:#recorro columna
        print(cell.value + " ",end="")#quito salto linea al mostrar valor
    print()#salto linea al mostrar fila
```

```
A B C
I D B
Z X V
```

14. FICHEROS . OPENPYXL

Añadimos al final.

```
import openpyxl

excel_document = openpyxl.load_workbook('C:\\EjerciciosPython\\Modulos\\EjemploExcel.xlsx')

Hoja = excel_document['Letras']#Selecciono la hoja de excel que deseo

Hoja.append(["J", "J", "G", "H"])

excel_document.save('C:\\EjerciciosPython\\Modulos\\EjemploExcel.xlsx')#Solo en este momento se modifica el fichero
```

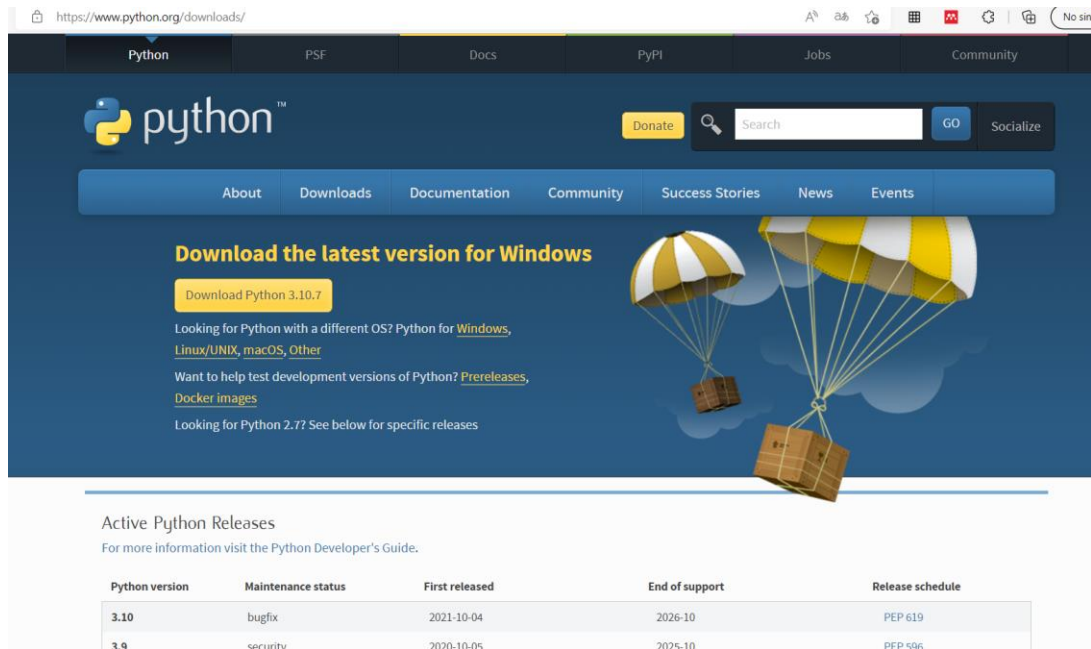
	A	B	C	D
1	A	B	C	
2	J	D	B	
3	Z	X	V	
4	J	J	G	H
5				
6				

...ahora...comencemos!

¡A programar!

1. Crea un **módulo en Python** que permita obtener de forma aleatoria **notas enteras entre 1 y 10**. (utiliza la librería Random).
2. Utilizando el módulo anterior, genera **30 notas aleatorias** y guárdalas en un archivo **Excel**
3. Crea un **gráfico** que muestre si hay más **aprobados** (notas ≥ 5) o **suspensos** (notas < 5).
4. Avanzado. Investiga la librería **Pandas** en Python.
Instálala y utilízala para trabajar con el archivo de **Excel** creado anteriormente.

ANEXO: INSTALAR PYTHON



The screenshot shows the Python.org website's download page. The header includes the Python logo, a search bar, and navigation links like 'About', 'Downloads', 'Documentation', 'Community', 'Success Stories', 'News', and 'Events'. The main content area features a large heading 'Download the latest version for Windows' and a button 'Download Python 3.10.7'. Below this, there are links for other operating systems and development versions. A decorative illustration of two parachutes with cargo boxes is on the right. The bottom section, titled 'Active Python Releases', provides a table with details about the current and previous stable versions.

Download the latest version for Windows

[Download Python 3.10.7](#)

Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [macOS](#), [Other](#)

Want to help test development versions of Python? [Prereleases](#), [Docker images](#)

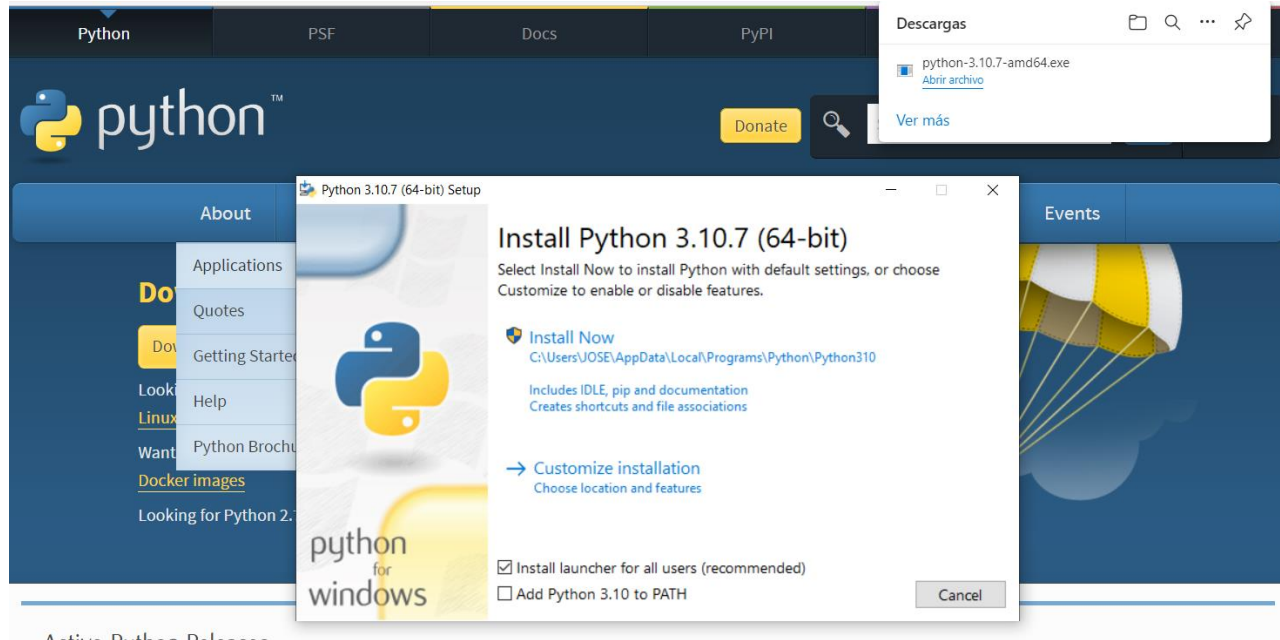
Looking for Python 2.7? See below for specific releases

Active Python Releases

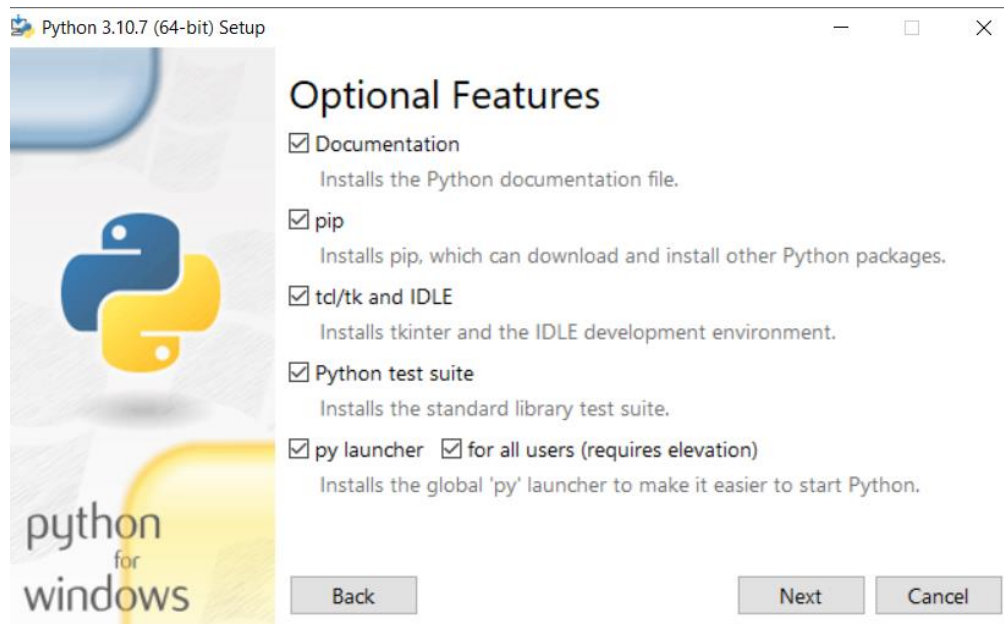
For more information visit the [Python Developer's Guide](#).

Python version	Maintenance status	First released	End of support	Release schedule
3.10	bugfix	2021-10-04	2026-10	PEP 619
3.9	security	2020-10-05	2025-10	PEP 596

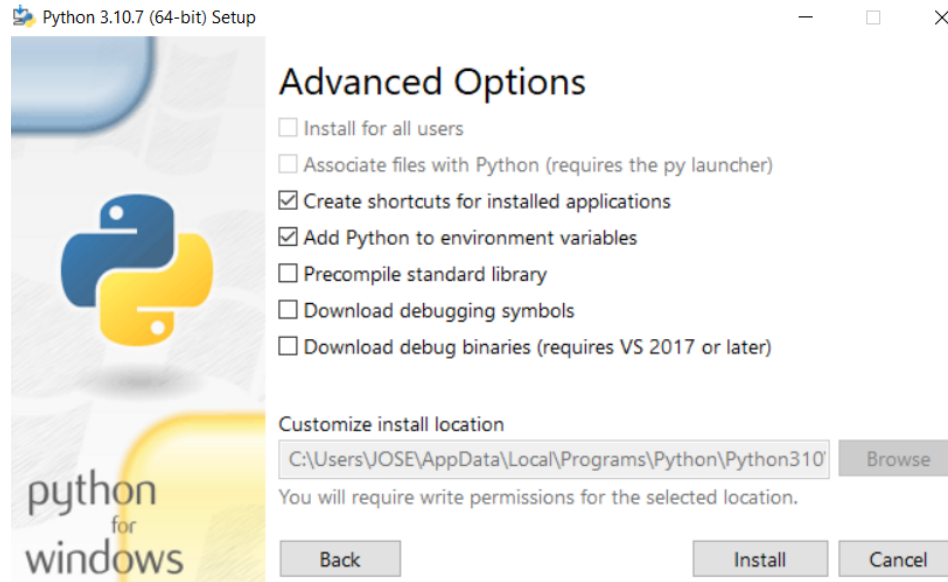
ANEXO: INSTALAR PYTHON



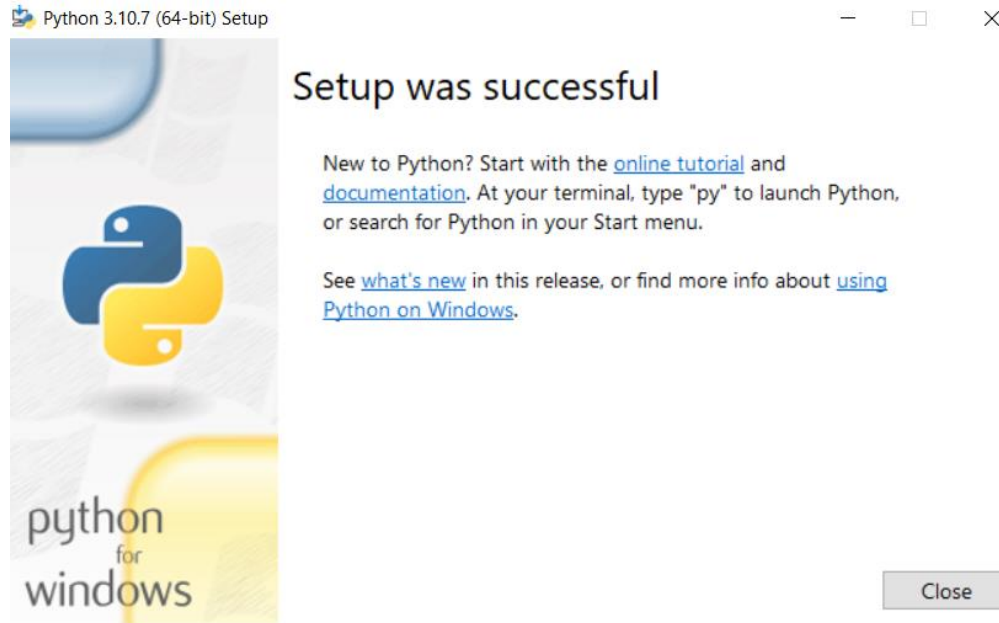
ANEXO: INSTALAR PYTHON



ANEXO: INSTALAR PYTHON



ANEXO: INSTALAR PYTHON



ANEXO: INSTALAR PYTHON

Administrador: Símbolo del sistema

```
C:\WINDOWS\system32>pip install openpyxl
Collecting openpyxl
  Downloading openpyxl-3.0.10-py2.py3-none-any.whl (242 kB)
    ----- 242.1/242.1 kB 3.7 MB/s eta 0:00:00
Collecting et_xmlfile
  Downloading et_xmlfile-1.1.0-py3-none-any.whl (4.7 kB)
Installing collected packages: et_xmlfile, openpyxl
Successfully installed et_xmlfile-1.1.0 openpyxl-3.0.10

C:\WINDOWS\system32>
```

