

Lecciones Aprendidas Python

**V.1.0.2**

HOJA DE CONTROL

|  |  |  |  |
| --- | --- | --- | --- |
| **Organización** | -- | | |
| **Curso** | LECCIONES APRENDIDAS PYTHON | | |
| **Entregable** |  | | |
| **Autor** | RAÚL MARCELO ARMAS BENAVIDES | | |
| **Versión** | 1.0.0 | **Fecha Versión** | 02/04/2021 |
| **Aprobado por** |  | **Fecha Aprobación** | 02/04/2021 |
|  |  | **Nº Total de Páginas** | XX |

REGISTRO DE CAMBIOS

|  |  |  |  |
| --- | --- | --- | --- |
| **Versión** | **Causa del Cambio** | **Responsable del Cambio** | **Fecha del Cambio** |
| 1.0.0 | L.A | RMAB | 02/04/2021 |
| 1.0.1 | L.A | RMAB | 15/07/2021 |
| 1.0.2 | L.A | RMAB | 25/08/2021 |

Contenido

[1 Tipo de datos 5](#_Toc80758703)

[1.1 Cadenas 5](#_Toc80758704)

[2 Programación orientada a objetos 6](#_Toc80758705)

[2.1 Conceptos 6](#_Toc80758706)

[2.1.1 Poliformismo 6](#_Toc80758707)

[2.2 Funciones 6](#_Toc80758708)

[2.3 Paquetes 6](#_Toc80758709)

[2.3.1 Enquiries (parece que está obsoleta) 6](#_Toc80758710)

[2.3.2 Pandas 7](#_Toc80758711)

[3 Manejo de ficheros 8](#_Toc80758712)

[3.1 Unicodeescape 8](#_Toc80758713)

[4 PYINSTALLER 9](#_Toc80758714)

[4.1 Instalación 10](#_Toc80758715)

[4.2 Primer ejecutable 10](#_Toc80758716)

[4.3 Ejecutable con interfaz 10](#_Toc80758717)

[4.4 Ejecutable en un fichero 10](#_Toc80758718)

[4.5 Cambiar el icono 11](#_Toc80758719)

[4.6 Limitaciones 11](#_Toc80758720)

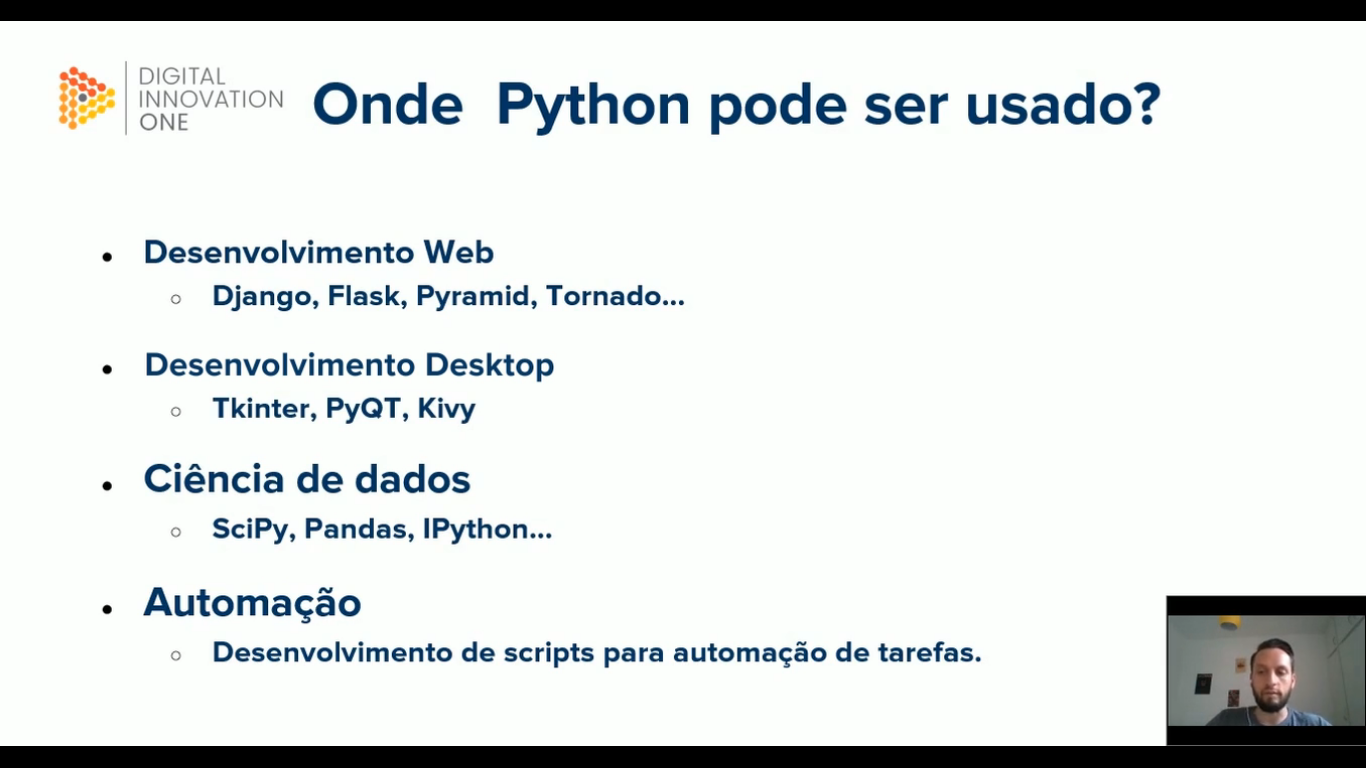
[5 Funciones lambda 11](#_Toc80758721)

[6 Conexión a base de datos 12](#_Toc80758722)

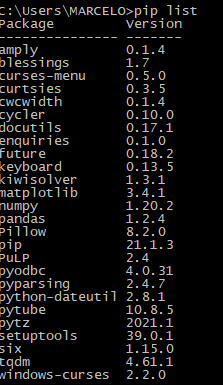
[7 Desarrollo web 12](#_Toc80758723)

[8 Ciencia de datos 12](#_Toc80758724)

[8.1 Regresión lineal 12](#_Toc80758725)



Para saber qué paquetes tengo instalado



Para ver el detalle de algún paquete determinado:

python -m pip show **numpy**

# Tipo de datos

## Cadenas

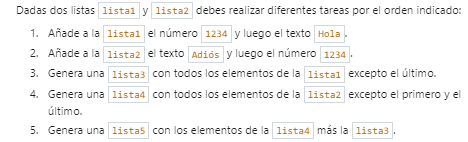
cadena\_corrupta = "airotsiH,6.7,aícraG nómaR"

cadena\_volteada = cadena\_corrupta[::-1]

# no modifiques el nombre de ninguna variable, puedes crear nuevas

lista1 = [1, 12, 123]

lista2 = ["Bye", "Ciao", "Agur", "Adieu"]



# completa el ejercicio

lista1.append(1234)

lista1.append("Hola")

lista2.append("Adiós")

lista2.append(1234)

lista3 = lista1[:-1]

lista4 = lista2[1:-1]

lista5 = lista4 + lista3

Teniendo una lista , cómo eliminamos los elemenos duplicados ¿ =??

**Lista - > conjunto y luego a ese conjutno lo transformamos a lista**

# Programación orientada a objetos

## Conceptos

### Poliformismo

Propiedad de la herencia en que objetos de distintas subclases pueden responder una misma acción.

En python todas las clases derivan de la superclase object, es decir, son polimórficas por defecto.

## Funciones

La sintaxis para una definición de función en Python es:

**def** NOMBRE(LISTA\_DE\_PARAMETROS):

*"""DOCSTRING\_DE\_FUNCION"""*

SENTENCIAS

RETURN [EXPRESION]

## Paquetes

### Enquiries (parece que está obsoleta)

the above process makes code messier and requires root privileges to run on Linux. the best way is to use **enquiries**

pip3 install enquiries

and use the following code

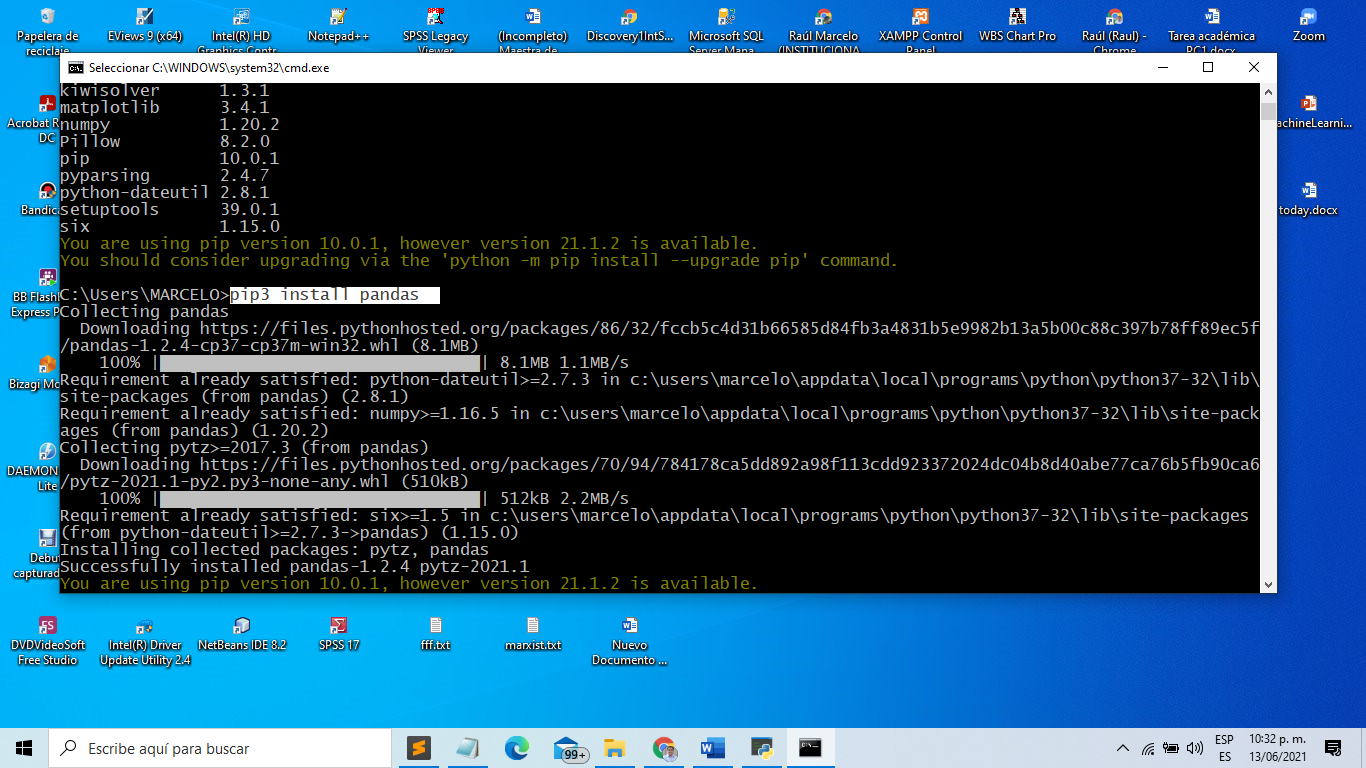
import enquiries

options = ['Do Something 1', 'Do Something 2', 'Do Something 3']

choice = enquiries.choose('Choose one of these options: ', options)

print(choice)

### Pandas



**Problem:**

I installed numpy following this [link](http://sourceforge.net/projects/numpy/files/NumPy/1.3.0/numpy-1.3.0.win-amd64-py2.6.msi/download?use_mirror=ufpr) - as suggested in the question. The installation went fine but when I execute

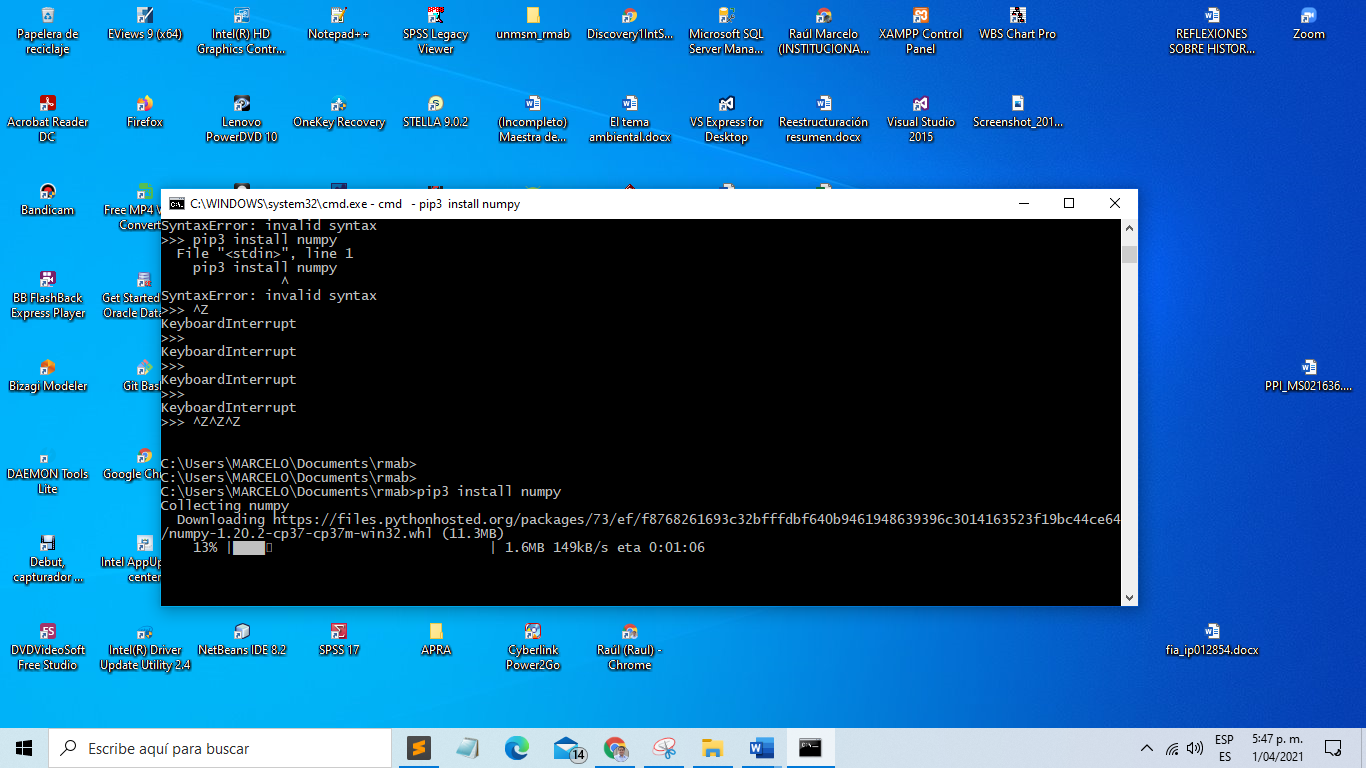
import numpy

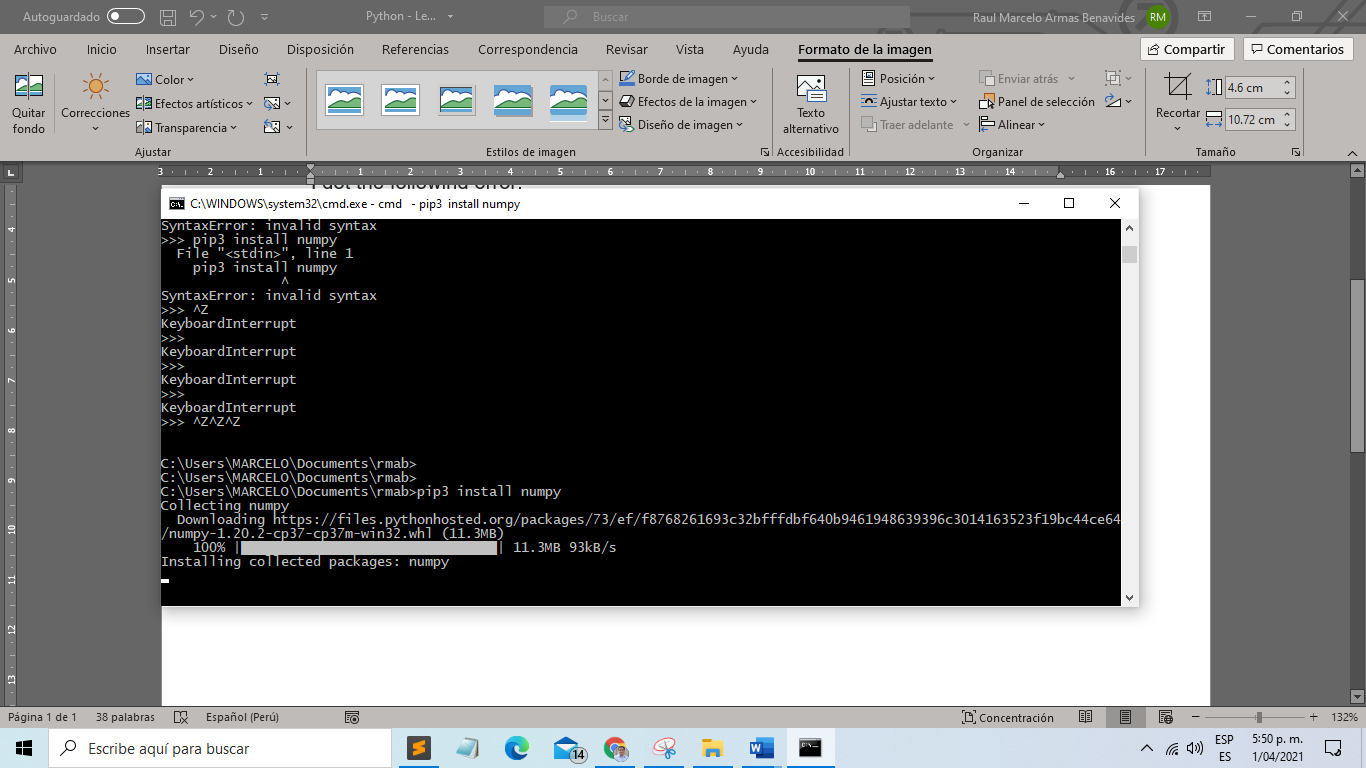
I got the following error:

Import error: No module named numpy

**Solution:**

Pip3 install numpy





# Manejo de ficheros

## Unicodeescape

SyntaxError: (unicode error) 'unicodeescape' codec can't decode bytes in position 2-3: truncated \UXXXXXXXX escape

[Finished in 552ms]

This error occurs because you are using a normal string as a path. You can use one of the three following solutions to fix your problem:

1: Just put r before your normal string it converts normal string to raw string:

pandas.read\_csv(r"C:\Users\DeePak\Desktop\myac.csv")

2:

pandas.read\_csv("C:/Users/DeePak/Desktop/myac.csv")

3:

pandas.read\_csv("C:\\Users\\DeePak\\Desktop\\myac.csv")

[Share](https://stackoverflow.com/a/46011113)

Follow

[edited Oct 23 '20 at 14:36](https://stackoverflow.com/posts/46011113/revisions)

[[](https://stackoverflow.com/users/6116066/ncw)](https://stackoverflow.com/users/6116066/ncw)

[ncw](https://stackoverflow.com/users/6116066/ncw)

**1,370**11 gold badge1212 silver badges2222 bronze badges

answered Sep 2 '17 at 6:27

[DeePak M. Birajdar](https://stackoverflow.com/users/6764110/deepak-m-birajdar)

**4,559**11 gold badge66 silver badges22 bronze badges

* 9

I like the 2nd option, it makes path portable across Windows and Linux. Thanks for Python's shielding the peculiarity of Windows. – [Yu Shen](https://stackoverflow.com/users/126164/yu-shen) [Apr 24 '18 at 16:15](https://stackoverflow.com/questions/37400974/unicode-error-unicodeescape-codec-cant-decode-bytes-in-position-2-3-trunca#comment87029346_46011113)

# PYINSTALLER

Ya hemos visto como distribuir nuestros paquetes... ¿Pero y si creamos una aplicación y queremos generar un ejecutable para utilizarla? Bueno, en este caso puede ser bastante complicado dependiendo de las dependencias que utilice el programa.

Por suerte hay un módulo que nos ayudará mucho a generar ejecutables porque automatiza el proceso, ese es **pyinstaller**.

Lo que hace es generar un .EXE en Windows, un .DMG en MAC o el ejecutable que utilice el sistema operativo. Dentro del ejecutable se incluye el propio intérprete de Python, y por esa razón podremos utilizarlo en cualquier ordenador sin necesidad de instalar Python previamente.

## Instalación

La instalación es muy fácil:

pip install pyinstaller

No hay más.

## Primer ejecutable

Comencemos con algo simple, tenemos un script **hola.py**:

print("Hola mundo!")

Y queremos crear un ejecutable a partir de él, pues haríamos lo siguiente:

pyinstaller hola.py

Una vez acabe el proceso se nos habrán creado varias carpetas. La que nos interesa es **dist**, y dentro encontraremos una carpeta con el nombre programa y en esta un montón de ficheros y el ejecutable, en mi caso como estoy en Windows es **hola.exe**.

Como es un programa para terminal, para ejecutarlo tengo que abrir la terminal en ese directorio y ejecutar el programa manualmente:

C:\Users\Hector\Desktop\hola\dist\hola>hola.exe

Hola mundo!

## Ejecutable con interfaz

Ahora vamos a hacer otro a partir de un simple programa con Tkinter, la librería de componentes integrada en Python que ya conocemos. Nos debería funcionar sin problemas:

from tkinter import \*

root = Tk()

Label(text='Hola mundo').pack(pady=10)

root.mainloop()

Suponiendo que lo hemos puesto en el mismo script:

pyinstaller hola.py

En esta ocasión si ejecutamos el programa con doble clic nos funcionará bien, el problema es que se muestra la terminal de fondo.

Para que desaparezca tenemos que indicar que es una aplicación en ventana, y eso lo hacemos de la siguiente forma al crear el ejecutable:

pyinstaller --windowed hola.py

## Ejecutable en un fichero

Ya véis que por defecto Pyinstaller crea un directorio con un montón de ficheros. Podemos utilizar un comando para generar un solo fichero ejecutable que lo contenga todo, pero este ocupara bastante más:

pyinstaller --windowed --onefile hola.py

## Cambiar el icono

También podemos cambiar el icono por defecto del ejecutable. Para ello necesitamos una imagen en formato .ico.

pyinstaller --windowed --onefile --icon=./hola.ico hola.py

Si por algo no os cambia el icono, probad cambiando el nombre del ejecutable. A veces el caché de Windows puede ignorar estas cosas.

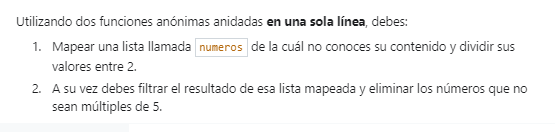
## Limitaciones

El gran problema con Pyinstaller como os decía al principio son las dependencias.

Si nuestro programa utiliza únicamente módulos de la librería estándard no tendremos ningún problema, pero si queremos utilizar módulos externos es posible que no funcione... A no ser que sea alguno de los soportados como PyQT, django, pandas, matpotlib... pero requieren una configuraciones extra.

Si queréis saber más os dejo [este enlace con los paquetes soportados](https://github.com/pyinstaller/pyinstaller/wiki/Supported-Packages).

# Funciones lambda



from evaluate import numeros

# Completa el ejercicio en la siguiente línea

numeros =list(filter(lambda n:n%5==0,map(lambda n:n/2,numeros)))

# Pruebas

Textfixture

Setup

TearDown

# Conexión a base de datos

# Desarrollo web

Flask

# Ciencia de datos

## Regresión lineal

\_\_future\_\_

Builtins

Numpy

matplotlib