

***Bienvenidos al Curso
Programación I !!***

Maestría en Ciencia de Datos

Universidad de Guadalajara

IT PhD Rocío del Carmen Chávez Álvarez

Contenido del Programa

Presentación

El maestrante en Ciencia de Datos debe tener bases sólidas de programación que le permitan desarrollar sus propios sistemas de análisis de datos, a la par de ser capaz de extraer la información de diferentes repositorios (bases de datos y/o archivos), gestionar su código con sistemas de control de versiones (Git), y probar y depurar su propio código

Con este curso, el futuro maestro en Ciencia de Datos, obtendrá las habilidades necesarias para dominar el lenguaje de programación Python

Objetivo General

Adquirir las bases del lenguaje de programación Python que le permitan al alumno desarrollar y probar cualquier tipo de programas en Python, pudiendo hacerlo de manera colaborativa

Temario

- Unidad 1 Arquitectura de Python
- Unidad 2 Fundamentos de Programación en Python
- Unidad 3 Funciones de Cadena
- Unidad 4 Archivos
- Unidad 5 Estructuras de Datos en Python
- Unidad 6 Expresiones Regulares
- Unidad 7 Sockets y Servicios Web
- Unidad 8 Bases de Datos SQL
- Unidad 9 Herramientas para depuración y pruebas de Código
- Unidad 10 Gestión de Repositorios de Código para control de versiones y desarrollo colaborativo con Git y GitHub

Evaluación

- Ejercicios Prácticos en Python 60%
- Ejercicios intermedios de Python 25%
- Ejercicio de Git 15%

Arquitectura de Python

Arquitectura de Python

Tipos de Lenguajes de Programación

Características de Python

Usos de Python

Historia

Algunas Implementaciones

Versiones de Python

IDEs

Tipos de lenguajes

Se clasifican con base en:

Su compilación

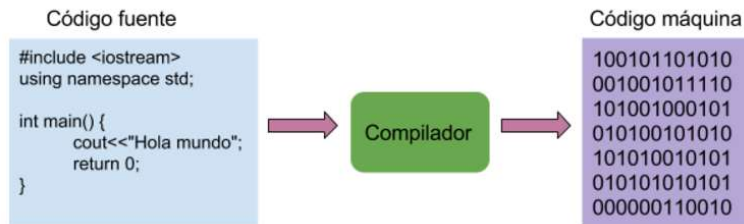
Necesidad de indicarle el tipo de variables a utilizar

Flexibilidad

Nivel de interpretabilidad

Tipos de lenguajes según su Compilación

Compilado: Primero se genera el script y después es traducido a un lenguaje que comprenda la máquina para crear un archivo ejecutable.



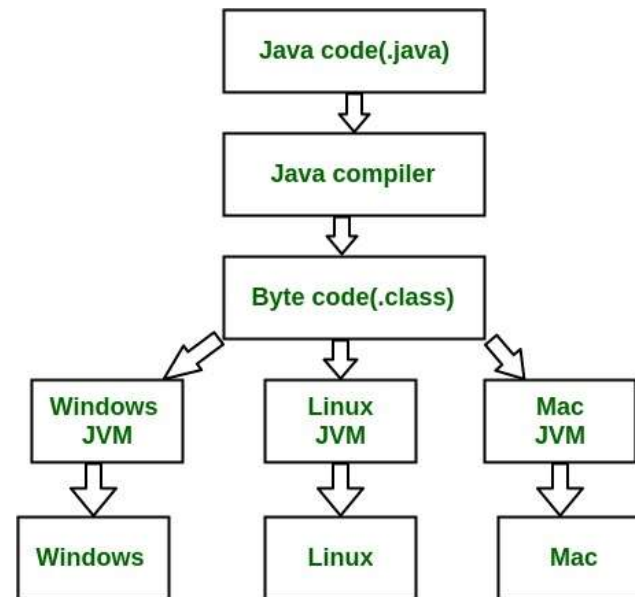
- Es necesario esperar a que el script sea compilado para saber si existen errores en el código
- Una vez compilados suelen ejecutarse más rápidamente

Ejemplos: C y C++

Interpretado: Requiere de un intérprete que va traduciendo el código al momento de irlo ejecutando.
Ejemplos: Python y lenguaje R

Intermedio: Se compila el código fuente y luego se ejecuta dependiendo de la máquina en la que se esté ejecutando
Ejemplo: Java

Ejemplo de Lenguaje Intermedio



Lenguajes Tipados y No Tipados

Lenguaje Tipado o Fuertemente Tipado: Tiene un sistema que define a qué tipo de dato pertenecen los valores y las expresiones utilizadas, así como la manera en la que los tipos se pueden manipular y como pueden interactuar.

Debido a que cualquier valor simplemente consiste en un conjunto de bits de un ordenador, el hardware no hace distinción entre caracteres, enteros y números en coma flotante o cualquier otro tipo de dato, por lo que es necesario informar a la computadora cómo deben ser tratados los bits utilizados para almacenar los datos.

Por ejemplo en C++

```
int f = 5
```

Lenguaje No Tipado o Débilmente Tipado: Si en una variable almaceno un número y después en esa misma variable almaceno un carácter, no hay problema.

Python pertenece a este tipo de lenguajes

Lenguajes Estáticos y Dinámicos

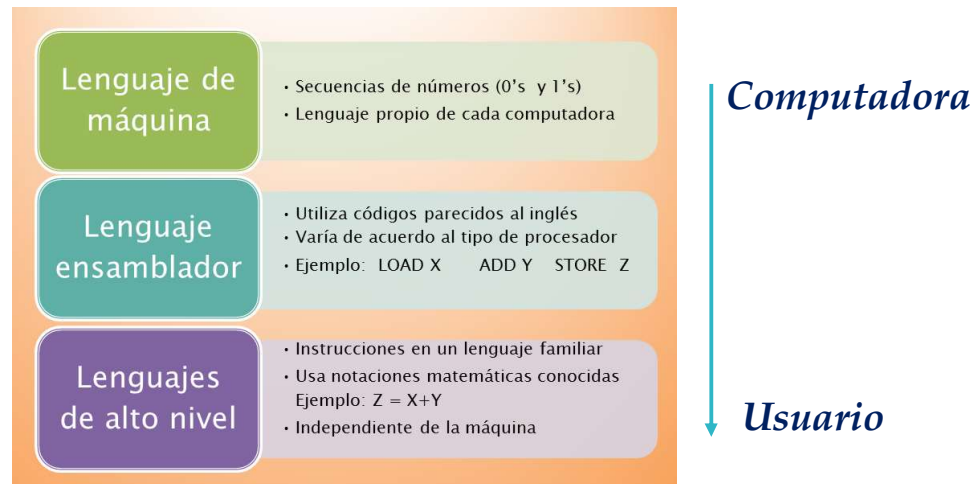
Lenguaje Estático: El tipo de una variable no puede cambiar una vez habiendo sido declarado ya que la comprobación del tipo se lleva a cabo al momento de la compilación

Lenguaje dinámico: El intérprete asigna el tipo a las variables durante el tiempo de ejecución basado en el valor que tiene en ese momento

Lenguajes de Alto y Bajo Nivel

Lenguaje de Alto Nivel: Utilizan instrucciones que son fácilmente comprendidos por el ser humano, ya que son muy parecidas al idioma inglés

Lenguaje de Bajo Nivel: Contiene instrucciones básicas reconocibles solamente por la computadora



Características de Python

- Fácil de aprender
- Fácil de leer
- Desarrollo de programas en poco tiempo
- Interactivo
- Existen muchas librerías útiles para diferentes tipos de desarrollos
- Interpretado, no es necesario compilar el código. (Lenguaje de Scripting)
- Debido a que no es compilado, no es veloz
- De código abierto
- Utilizado en muchas ocasiones para construir prototipos

C++

```
#include <iostream>

using namespace std;

int main()
{
    cout << "Hello world!" << endl;
    return 0;
}
```

Java

```
public class HelloWorld {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("Hello world!");
    }

}
```

Python

```
print("Hello World")
```

Tiempo promedio requerido para crear un programa



Fuente: Prechelt & Garret, tiempo total necesario para producir un programa de búsqueda/procesado de texto usando diferentes lenguajes.

Usos de Python

- Aplicaciones Web
- Ciberseguridad
- Desarrollo Back End
- Extracción de Información de sitios web (Web Scraping)
- Automatización
- Desarrollo de juegos
- Análisis de Datos
- Machine Learning
- Deep Learning
- IoT

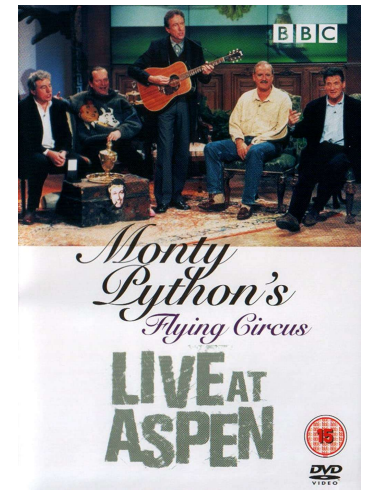


Algunas Empresas que Utilizan Python



Historia de Python

- Creado por Guido Van Rossum a finales de los 80's y principios de los 90's
- Está basado en otro lenguaje llamado ABC desarrollado por la compañía para que trabajaba Guido Van Rossum en los años 80's
- Lo llamó Python en honor a un grupo de cómicos famosos llamado Monty Python
- Versiones:
 - 0.9.0 febrero de 1991
 - 1.0 en 1994
 - 2.0 en octubre del año 2000
 - 3.0 en Diciembre del 2008
- Se encuentran vigentes las versiones 2 y 3, las cuales son incompatibles
- En la actualidad existen conferencias anuales llamadas PyCon



ABC vs. Python

Recuperar las palabras de un documento en ABC

```
HOW TO RETURN words document:
  PUT {} IN collection
  FOR line IN document:
    FOR word IN split line:
      IF word not.in collection:
        INSERT word IN collection
  RETURN collection
```

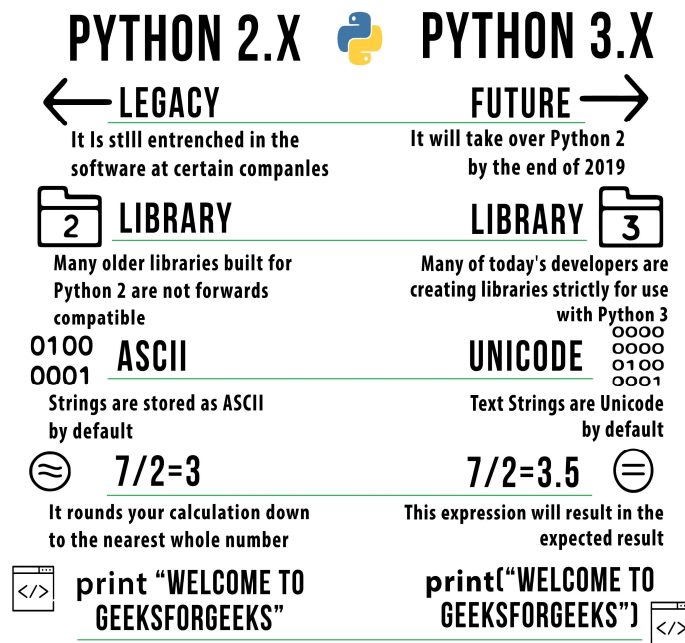
Recuperar las palabras de un documento en Python

```
def words(document):
    collection = set()
    for line in document:
        for word in line.split():
            if word not in collection:
                collection.add(word)
    return collection
```

Versiones de Python

- La última versión de Python 2, fue la 2.7
- Python 3
- MicroPython Corre en microcontroladores
 - Raspberry Pi
 - Micro:bit
 - Calliope

Python 2 vs Python 3



Existen muchas diferencias en el código de ambas versiones, por lo que no pueden ser compatibles

Unicode tiene muchísimos caracteres en comparación con lo caracteres que contiene el código ASCII

En Python 2 la división arrojaba solamente el número entero del cociente, en Python 3, podemos obtener la cifra completa, incluyendo los decimales

En Python 2 print era una simple instrucción, en Python 3, print es una función

Algunas Implementaciones de Python

CPython El intérprete está escrito en lenguaje C



Jython



IronPython Soportado por .Net y Mono(para móviles),
las cuales son plataformas de Microsoft
destinadas a desarrollar aplicaciones



PyPy Permite ejecutar código mucho más rápido que Cpython



Cython Optimiza a Python, ya que lo traduce a C++



Cython

Python

print("Hello World")



C++

```
#include <iostream>
using namespace std;
int main()
{
    cout << "Hello World" <<
endl;
    return 0;
}
```

Ejemplo de Python vs. Cython

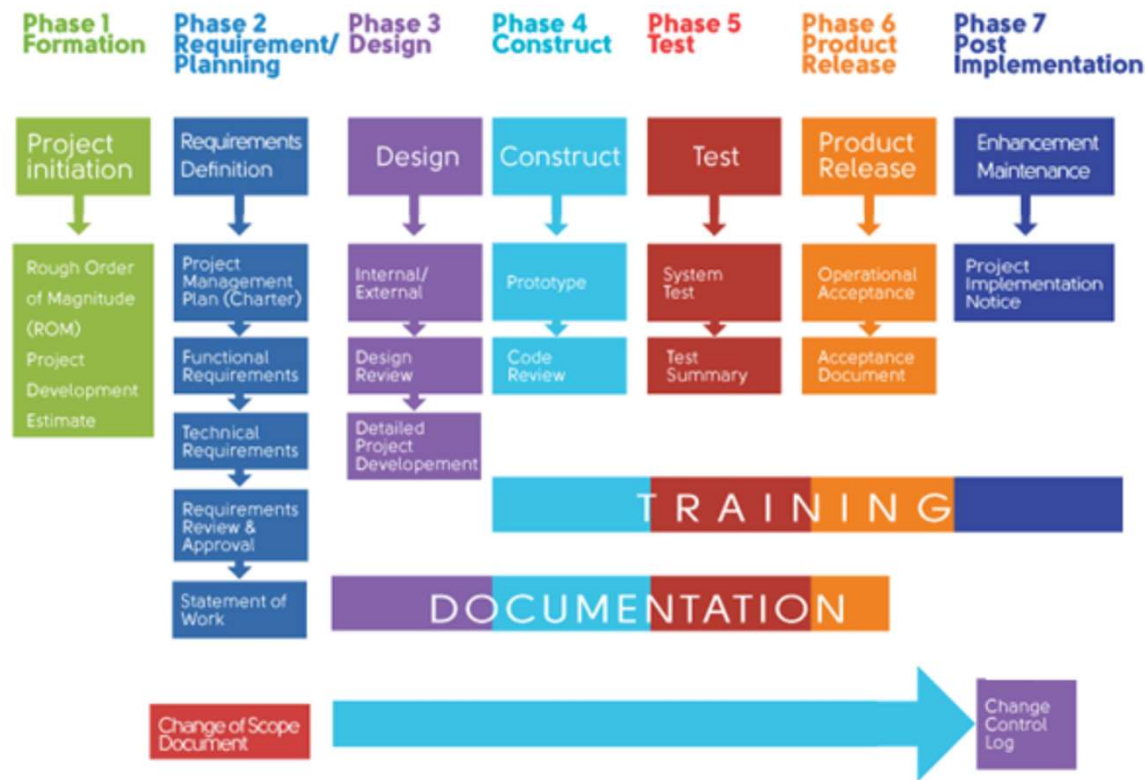
Python

```
def f(x):  
    return x**2-x  
  
def integrate_f(a, b, N):  
    s = 0  
    dx = (b-a)/N  
    for i in range(N):  
        s += f(a+i*dx)  
    return s * dx
```

Cython

```
cdef f(double x):  
    return x**2-x  
  
def integrate_f(double a, double b, int N):  
    cdef int i  
    cdef double s, x, dx  
    s = 0  
    dx = (b-a)/N  
    for i in range(N):  
        s += f(a+i*dx)  
    return s * dx
```

Fases de Desarrollo de Software



Pseudocódigo

- No contiene código
- Fácil de comprender
- Permite concentrarse en la solución del problema
- Facilita la detección de errores

Ejemplo de Pseudocódigo

Obtención del promedio de los alumnos y del conteo de los aprobados

```
Inicializar contador de aprobados a cero
Inicializar contador de estudiantes a uno
Inicializar variable calificación a cero
Obtener la cantidad de estudiantes
Mientras el contador de estudiantes sea menor o igual al total de estudiantes
    Agregar resultado del examen a la variable calificación
    Si el resultado del examen es mayor a 59
        Sumar 1 al contador de aprobados
        Incrementar en uno el contador de estudiantes
    Si no
        Incrementar en uno el contador de estudiantes
    Finaliza la sentencia if-else
Finaliza el ciclo while
Calcular el promedio
Imprimir en pantalla el promedio
Imprimir en pantalla la cantidad de aprobados
```

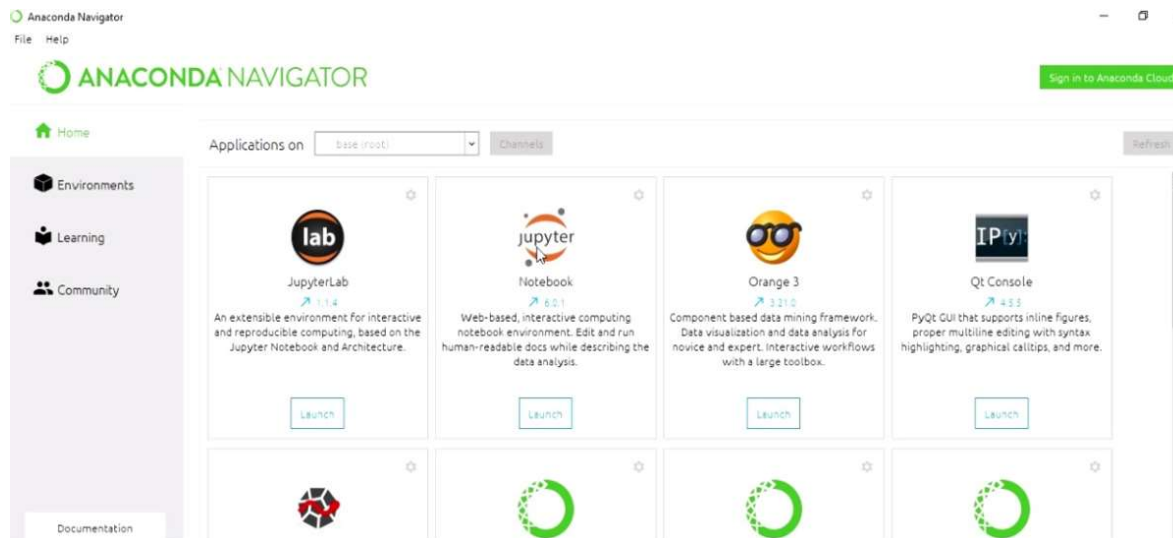
Nota: La indentación es muy importante en la claridad del pseudocódigo

IDEs para Python



IDE: Integrated Development Environment

Instalación de Anaconda



Ver instrucciones en el video **Instalación de Jupyter Notebook (Anaconda)**
https://www.youtube.com/watch?v=mOCy5bq4AYw&ab_channel=RocioChavezCienciadeDatos

Plan Propuesto de Trabajo

1

```
Anaconda Powershell Prompt
(base) PS C:\Users\rocio> python
Python 3.7.4 (default, Aug 9 2019, 18:34:13) [MSC v.1915 64 bit (AMD64)] ::
Anaconda custom (64-bit) on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```

2

```
Anaconda Powershell Prompt
(base) PS C:\Users\rocio>
```

```
C:\Users\rocio\Python_en_terminal\prueba2.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
< > prueba2.py x untitled
1 print("Hola, estoy haciendo pruebas para correr python desde la terminal")
2
3 print(10+5+7)
4
5 import numpy as np
6 import pandas as pd
```

3



Notebook

[6.4.3](#)

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

4



Spyder

[5.1.1](#)

Scientific Python Development Environment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Utilizando Python
desde la terminal de
Anaconda

Actualización de Paquetes

Desde el Anaconda Powershell Prompt

```
>conda update conda
```

```
>conda update --all
```

Nota: En windows es necesario abrir el prompt como administrador, dando clic en el botón derecho del mouse

Lista de paquetes instalados en Anconda

Anaconda Prompt

```
(base) C:\Users\roci>conda list
# packages in environment at C:\Users\roci\Anaconda3:
#
# Name                          Version          Build      Channel
_ipyw_jlab_nb_ext_conf         0.1.0            py37_0
absl-py                         0.7.1            pypi_0     pypi
alabaster                       0.7.12           py37_0
anaconda                        2019.10           py37_0
anaconda-client                 1.7.2            py37_0
anaconda-navigator              1.10.0           py37_0
anaconda-project                0.8.3            py_0       conda-forge
anyqt                           0.0.13           pyh6c4a22f_0 conda-forge
asn1crypto                      1.0.1            py37_0     conda-forge
astor                           0.8.0            pypi_0     pypi
astroid                         2.3.1            py37_0     conda-forge
astropy                         3.2.1            py37he774522_0
atomicwrites                    1.3.0            py37_1
attrs                           19.2.0           py_0       conda-forge
babel                           2.7.0            py_0       conda-forge
backcall                        0.1.0            py37_0
backports                       1.0              py_2       conda-forge
backports.os                    0.1.1            py37_0
backports.shutil_get_terminal_size 1.0.0            py37_2     conda-forge
beautifulsoup4                  4.8.0            py37_0
bitarray                        1.0.1            py37he774522_0
bkcharts                        0.2              py37_0
blas                            1.0              mkl
bleach                           3.1.0            py37_0
blosc                           1.16.3           h7bd577a_0
bokeh                           1.3.4            py37_0     conda-forge
boto                             2.49.0           py37_0
```


Instalando paquetes

```
(base) PS C:\Users\rocio> conda install matplotlib
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\rocio\Anaconda3

  added / updated specs:
    - matplotlib

The following packages will be downloaded:



| package                   | build          |         |             |
|---------------------------|----------------|---------|-------------|
| ca-certificates-2021.5.30 | h5b45459_0     | 171 KB  | conda-forge |
| certifi-2021.5.30         | py37h03978a9_0 | 142 KB  | conda-forge |
| matplotlib-3.3.2          | py37h03978a9_1 | 7 KB    | conda-forge |
| matplotlib-base-3.3.2     | py37h64d6e1c_1 | 6.8 MB  | conda-forge |
| openssl-1.1.1k            | h8ffe710_0     | 5.7 MB  | conda-forge |
| Total:                    |                | 12.8 MB |             |



The following NEW packages will be INSTALLED:

  matplotlib-base      conda-forge/win-64::matplotlib-base-3.3.2-py37h64d6e1c_1

The following packages will be UPDATED:

  ca-certificates      anaconda::ca-certificates-2020.10.14-0 --> conda-forge::ca-certificates-2021.5.30-h5b45459_0
  certifi              anaconda::certifi-2020.6.20-py37_0 --> conda-forge::certifi-2021.5.30-py37h03978a9_0
  matplotlib           pkgs/main::matplotlib-3.1.1-py37hc8f6~ --> conda-forge::matplotlib-3.3.2-py37h03978a9_1
  openssl              anaconda::openssl-1.1.1h-he774522_0 --> conda-forge::openssl-1.1.1k-h8ffe710_0

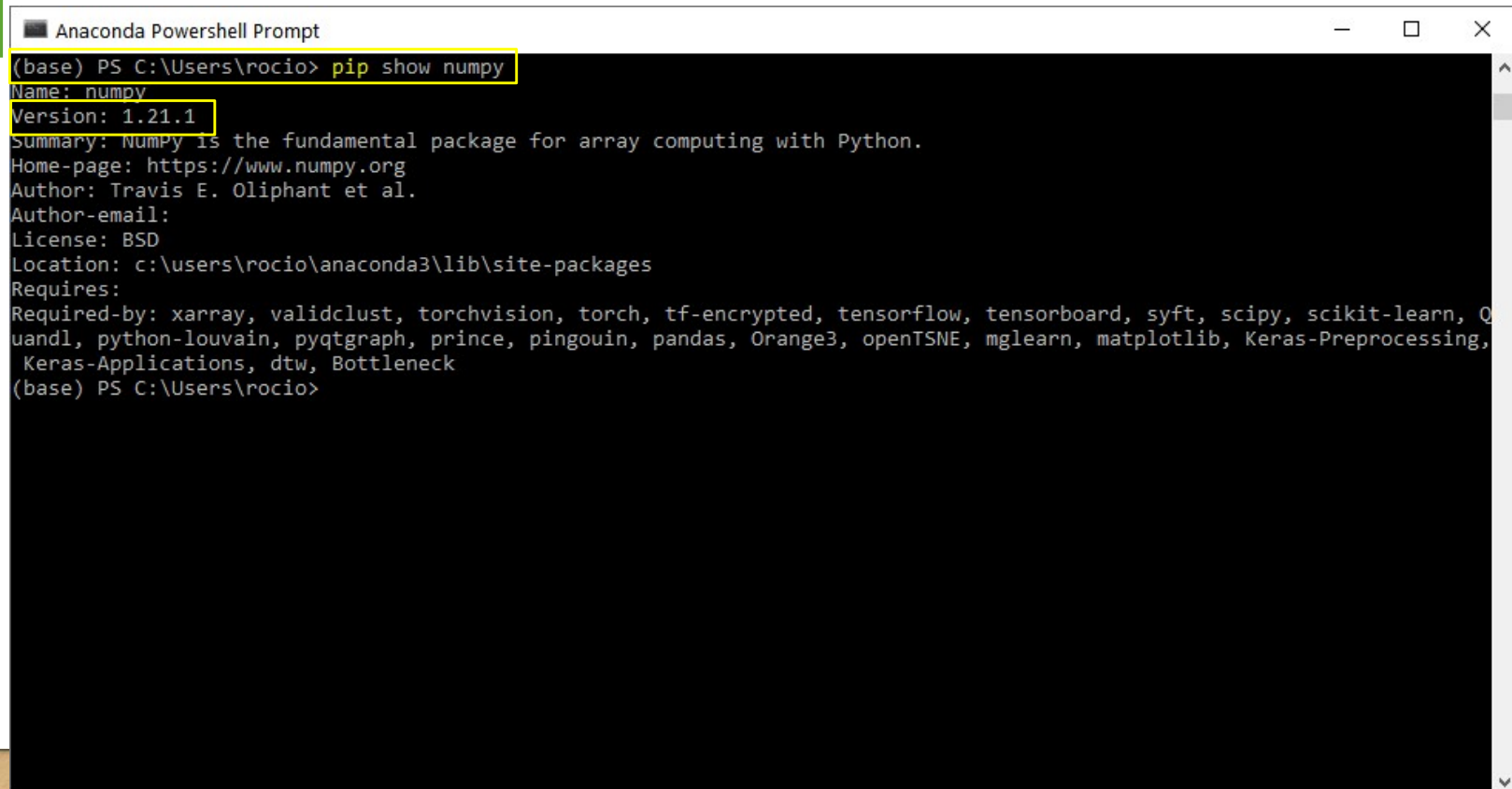
Proceed ([y]/n)? y

Downloading and Extracting Packages
openssl-1.1.1k          | 5.7 MB | #####
matplotlib-base-3.3.2  | 6.8 MB | #####
matplotlib-3.3.2       | 7 KB   | #####
certifi-2021.5.30      | 142 KB | #####
ca-certificates-2021   | 171 KB | #####
```

Para instalar una versión en específico:

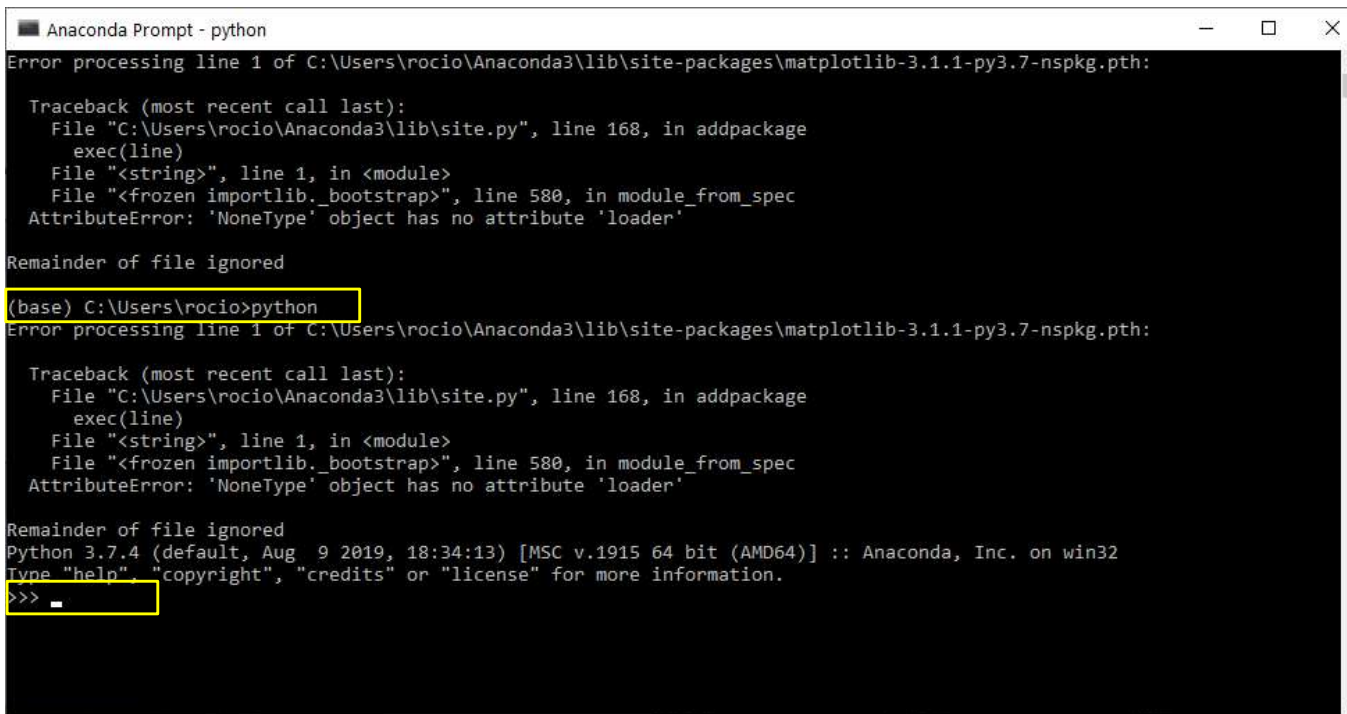
Ejemplo
`conda install matplotlib=3.1.1`

Consultar versión de Paquetes



```
Anaconda Powershell Prompt
(base) PS C:\Users\rocio> pip show numpy
Name: numpy
Version: 1.21.1
Summary: Numpy is the fundamental package for array computing with Python.
Home-page: https://www.numpy.org
Author: Travis E. Oliphant et al.
Author-email:
License: BSD
Location: c:\users\rocio\anaconda3\lib\site-packages
Requires:
Required-by: xarray, validclust, torchvision, torch, tf-encrypted, tensorflow, tensorboard, syft, scipy, scikit-learn, Q
uandl, python-louvain, pyqtgraph, prince, pinguin, pandas, Orange3, openTSNE, mglearn, matplotlib, Keras-Preprocessing,
Keras-Applications, dtw, Bottleneck
(base) PS C:\Users\rocio>
```

Utilizando el prompt de Anaconda



```
Anaconda Prompt - python
Error processing line 1 of C:\Users\rocio\Anaconda3\lib\site-packages\matplotlib-3.1.1-py3.7-nspkg.pth:

Traceback (most recent call last):
  File "C:\Users\rocio\Anaconda3\lib\site.py", line 168, in addpackage
    exec(line)
  File "<string>", line 1, in <module>
  File "<frozen importlib._bootstrap>", line 580, in module_from_spec
AttributeError: 'NoneType' object has no attribute 'loader'

Remainder of file ignored
(base) C:\Users\rocio>python
Error processing line 1 of C:\Users\rocio\Anaconda3\lib\site-packages\matplotlib-3.1.1-py3.7-nspkg.pth:

Traceback (most recent call last):
  File "C:\Users\rocio\Anaconda3\lib\site.py", line 168, in addpackage
    exec(line)
  File "<string>", line 1, in <module>
  File "<frozen importlib._bootstrap>", line 580, in module_from_spec
AttributeError: 'NoneType' object has no attribute 'loader'

Remainder of file ignored
Python 3.7.4 (default, Aug  9 2019, 18:34:13) [MSC v.1915 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```

Utilizar ctrl+z para salir de python

Ayuda en el prompt de Python

```
>>> help()
```

```
Welcome to Python 3.7's help utility!
```

Utilizar ctrl+c para salir de la ayuda

Anaconda Powershell Prompt

```
help> matplotlib
```

```
Help on package matplotlib:
```

NAME

```
matplotlib - An object-oriented plotting library.
```

DESCRIPTION

```
A procedural interface is provided by the companion pyplot module,
which may be imported directly, e.g.::
```

```
import matplotlib.pyplot as plt
```

```
or using ipython::
```

```
ipython
```

```
at your terminal, followed by::
```

```
In [1]: %matplotlib
```

```
In [2]: import matplotlib.pyplot as plt
```

```
at the ipython shell prompt.
```

```
For the most part, direct use of the object-oriented library is encouraged when
programming; pyplot is primarily for working interactively. The exceptions are
the pyplot functions `.pyplot.figure`, `.pyplot.subplot`, `.pyplot.subplots`,
and `.pyplot.savefig`, which can greatly simplify scripting.
```

Modules include:

```
:mod:`matplotlib.axes`
```

```
The `~.axes.Axes` class. Most pyplot functions are wrappers for
`~.axes.Axes` methods. The axes module is the highest level of OO
access to the library.
```

```
:mod:`matplotlib.figure`
```

```
The `.Figure` class.
```

```
:mod:`matplotlib.artist`
```

```
-- Más --
```

Limpiar el prompt de Python

```
import os
```

```
os.system("cls")
```

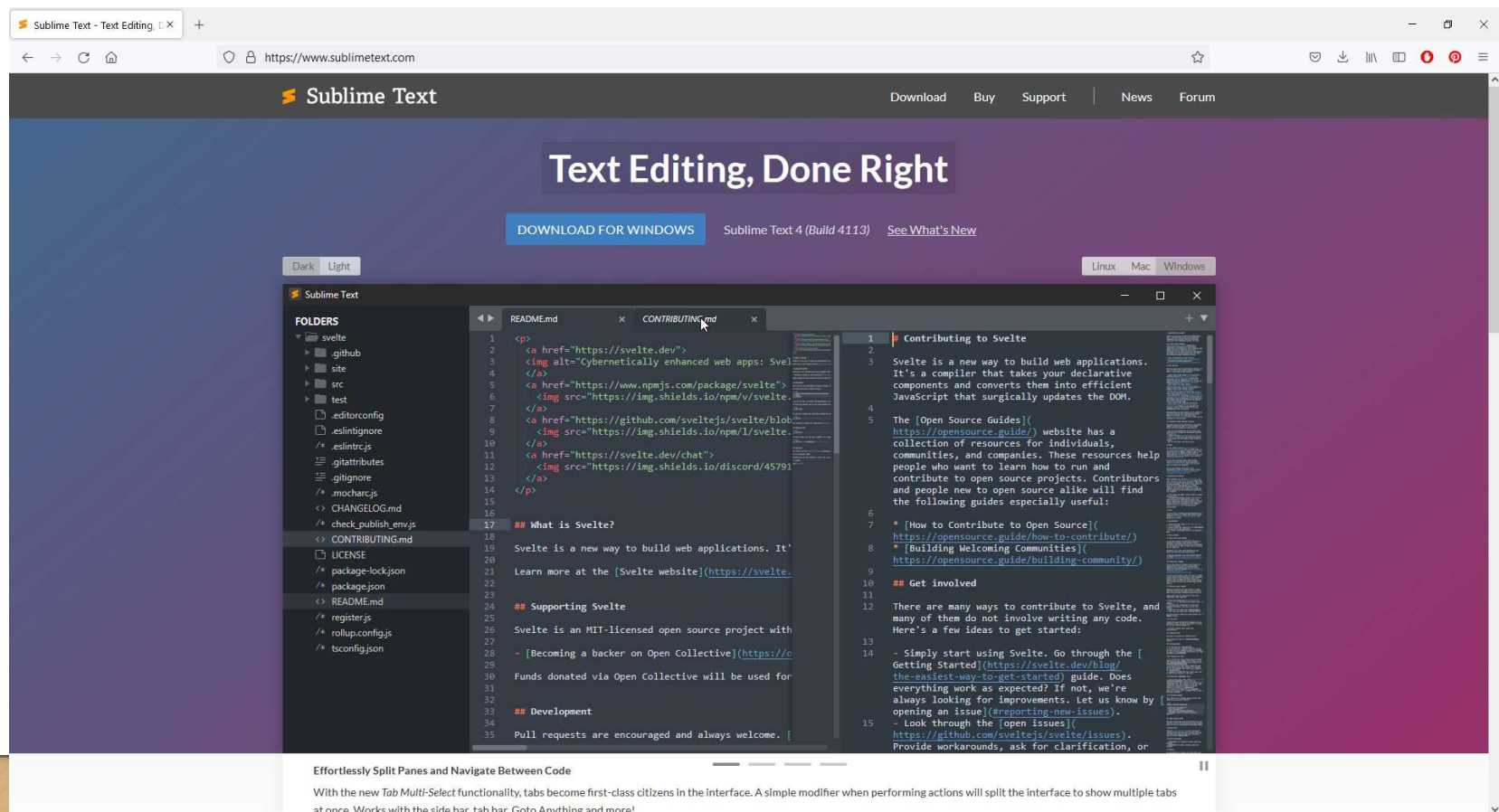
Ejecutando código en el prompt de Python



```

Anaconda Powershell Prompt
>>> 23/2
11.5
>>> 23//2
11
>>> _
```

Instalar Sublime text



Creación de un script en Sublime Text

```
C:\Users\rocio\Escritorio2\Pavillion-Rocio\Cursos impartidos Presenciales\Programacion I MCD\Scripts\Histogramas en Python.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

< > untitled Histogramas en Python.py x

1  #Importamos los paquetes que vamos a necesitar
2
3  import pandas as pd #Contiene funciones que nos ayudan en el análisis de datos
4  import matplotlib.pyplot as plt
5
6  #Leemos el archivo a analizar
7  atletas = pd.read_csv('C:/Users/rocio/Archivos en Jupyter/Bases de datos a analizar/categorias de corredores.csv',
8                        index_col=0) #Con index_col=0 le indicamos que las filas tienen un nombre
9  atletas.info()
10
11 #Vemos las primeras filas del archivo
12 atletas.head()
13
14 #Creamos el histograma de la variable Tiempo
15 plt.figure(1)
16 plt.hist(atletas['Tiempo'], 15, color="yellow", ec="black")
17 plt.title("Histograma Tiempo")
18
```


Ejecutando un script en el prompt de Anaconda

Opción 1

- En caso de que se hayan generado archivos al correr el script con anterioridad, es mejor borrarlos antes de crearlos de nuevo
- Copiamos todas las líneas del script
- Vamos al prompt de Python
- Damos clic en el botón derecho del mouse o ctrl v

```
Anaconda Powershell Prompt
>>> #Indicamos el directorio en el que vamos a trabajar
... import os
>>> os.chdir('C:/Users/rocio/Archivos en Jupyter/Bases de datos a analizar/')
>>>
>>> #Importamos los paquetes que vamos a necesitar
...
>>> import pandas as pd #Contiene funciones que nos ayudan en el análisis de datos
>>> import matplotlib.pyplot as plt
>>>
>>> #Leemos el archivo a analizar
... atletas = pd.read_csv('categorias de corredores.csv', index_col=0) #Con index_col=0 le indicam
>>> atletas.info()
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1000 entries, 3475 to 8527
Data columns (total 6 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Lugar       1000 non-null   int64
1   Genero       1000 non-null   object
2   Edad        1000 non-null   int64
3   Pais        999 non-null    object
4   Tiempo      1000 non-null   float64
5   Velocidad    1000 non-null   object
dtypes: float64(1), int64(2), object(3)
memory usage: 54.7+ KB
>>>
>>> #Vemos las primeras filas del archivo
... atletas.head()

```

	Lugar	Genero	Edad	Pais	Tiempo	Velocidad
Corredor						
3475	3592	Male	52	GBR	217.483333	Regular
13594	13853	Female	40	NY	272.550000	Regular
12012	12256	Male	31	FRA	265.283333	Regular
10236	10457	Female	33	MI	256.150000	Regular
9476	9686	Male	33	NY	252.250000	Regular

Ejecutando un script en el prompt de Anaconda

- Vamos al prompt de Anaconda
- Corremos el script con el comando “python” y el nombre del script entre comillas

```
Anaconda Powershell Prompt
(base) PS C:\Users\rocio> cd "C:\Users\rocio\Escritorio2\Pavillion-Rocio\Cursos impartidos Presenciales\Programacion I MCD\Scripts\"
(base) PS C:\Users\rocio\Escritorio2\Pavillion-Rocio\Cursos impartidos Presenciales\Programacion I MCD\Scripts> dir

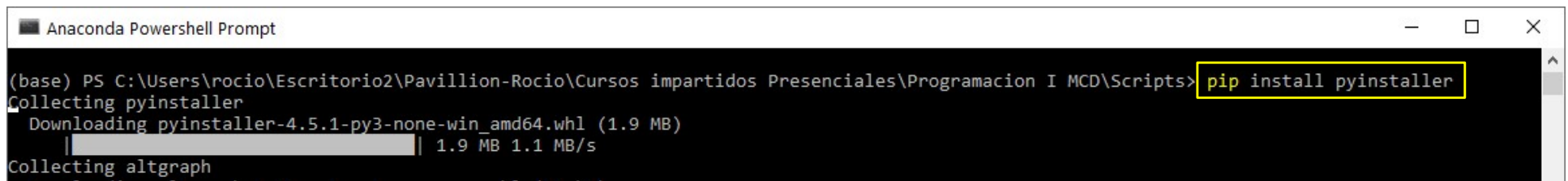
Directorio: C:\Users\rocio\Escritorio2\Pavillion-Rocio\Cursos impartidos Presenciales\Programacion I MCD\Scripts

Mode                LastWriteTime         Length Name
----                -
-a----            8/15/2021   6:43 PM          29555 Histograma.jpg
-a----            8/15/2021  11:27 PM           1322 Histogramas en Python.py
-a----            8/15/2021   6:34 PM            652 Python en terminal.py

(base) PS C:\Users\rocio\Escritorio2\Pavillion-Rocio\Cursos impartidos Presenciales\Programacion I MCD\Scripts> python "Histogramas en Python.py"
```

Creando un Archivo Ejecutable en el prompt de Anaconda

- Vamos al prompt de Anaconda
- Instalamos el paquete pyinstaller con el comando
pip install pyinstaller



```
Anaconda Powershell Prompt

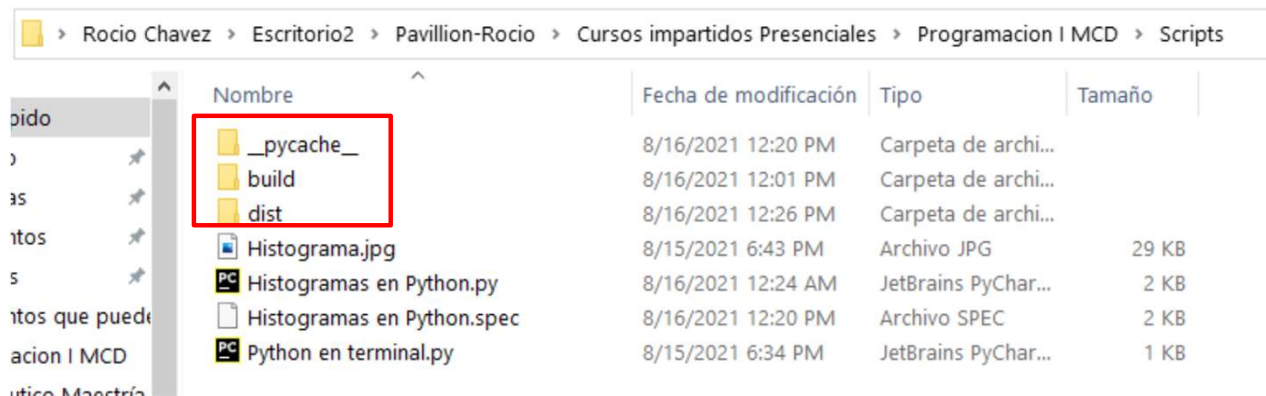
(base) PS C:\Users\rocio\Escritorio2\Pavillion-Rocio\Cursos impartidos Presenciales\Programacion I MCD\Scripts> pip install pyinstaller
Collecting pyinstaller
  Downloading pyinstaller-4.5.1-py3-none-win_amd64.whl (1.9 MB)
    | 1.9 MB 1.1 MB/s
Collecting altgraph
```

Creando un Archivo Ejecutable en el prompt de Anaconda (Cont.)

- Dentro de la carpeta en donde se encuentra el archivo que queremos convertir a ejecutable, tecleamos

pyinstaller --onefile nombre_del_archivo.py

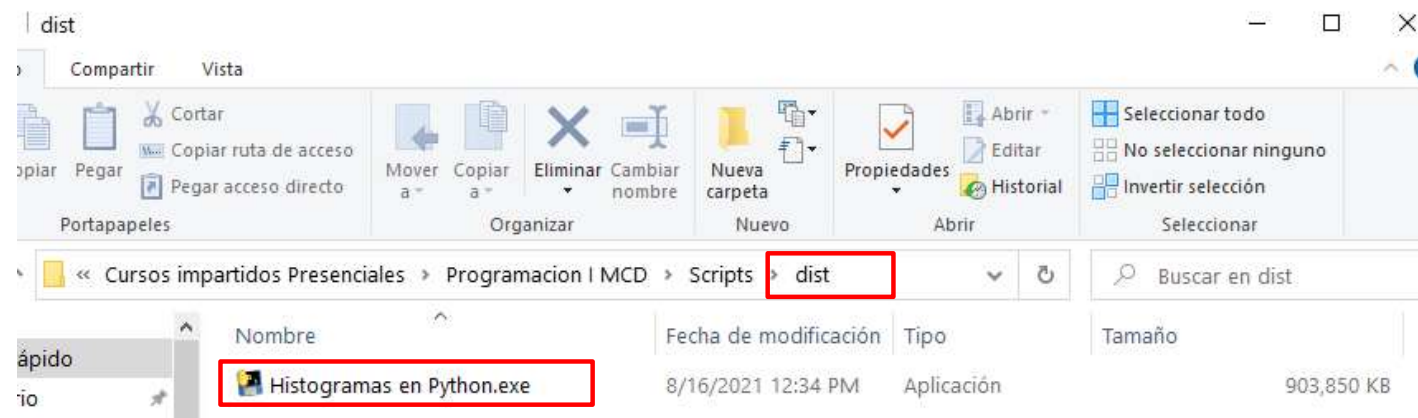
```
Anaconda Powershell Prompt
(base) PS C:\Users\rocio\Escritorio2\Pavillion-Rocio\Cursos impartidos Presenciales\Programacion I MCD\Scripts> pyinstaller --onefile "Histogramas en Python.py"
```



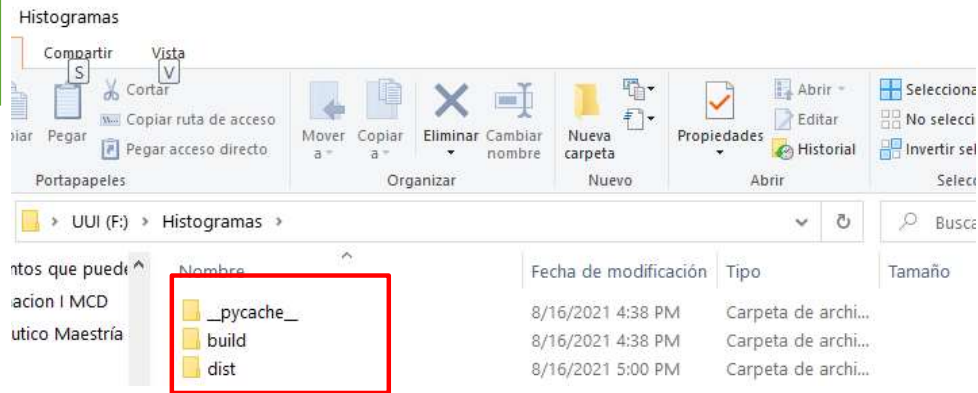
Nombre	Fecha de modificación	Tipo	Tamaño
__pycache__	8/16/2021 12:20 PM	Carpeta de archi...	
build	8/16/2021 12:01 PM	Carpeta de archi...	
dist	8/16/2021 12:26 PM	Carpeta de archi...	
Histograma.jpg	8/15/2021 6:43 PM	Archivo JPG	29 KB
Histogramas en Python.py	8/16/2021 12:24 AM	JetBrains PyChar...	2 KB
Histogramas en Python.spec	8/16/2021 12:20 PM	Archivo SPEC	2 KB
Python en terminal.py	8/15/2021 6:34 PM	JetBrains PyChar...	1 KB

Se crean las carpetas
build, dist y __pycache__

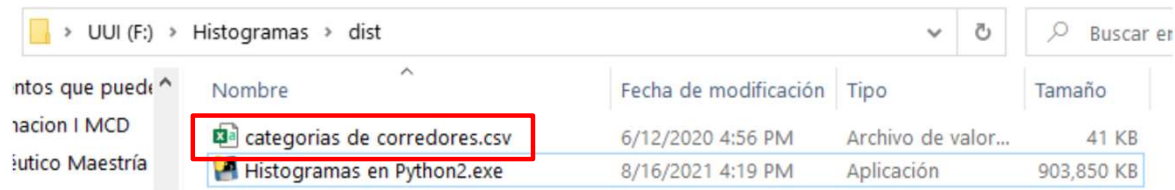
Ubicación del Archivo Ejecutable



Ubicación del Archivo Ejecutable en una USB



Nota importante: Es necesario colocar el archivo .csv en la misma carpeta en la que se encuentra el ejecutable



Tarea 1

- Crear un archivo ejecutable que realice lo siguiente:
 - Lea un archivo propio
 - Obtenga el histograma de una de las variables
 - Muestre el gráfico
 - Lo guarde en un archivo de tipo jpg
- El ejecutable deberá llevar tu nombre y apellido
- Subir a Google Drive:
 - El ejecutable
 - El archivo .csv que analizaste
 - El script .py del ejecutable
 - Un documento en Word con la captura de las pantallas del comando que utilizaste para crear el ejecutable
- Subir el link de Google Drive al classroom para que yo pueda revisar tu tarea