



INTRODUCCIÓN A LA  
PROGRAMACIÓN EN PYTHON PARA  
APLICACIONES AGROFORESTALES DE  
UNA MANERA FÁCIL Y AMENA

Día 1

14/02/2022



# ANDRÉS PEÑUELA FERNÁNDEZ

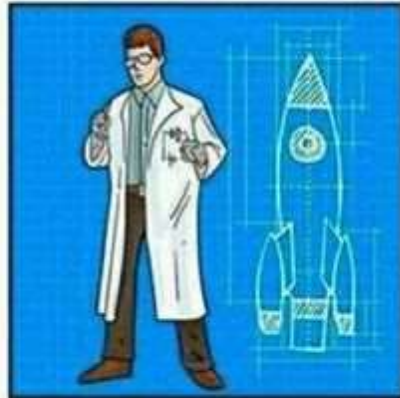
- Ingeniero de Caminos, Canales y Puertos
- Investigador posdoctoral en el departamento de Agronomía de la UCO
- +10 años de experiencia en programación de modelos hidrológicos y de erosión de suelo
- Lenguajes que he usado: Matlab y Python (y un poco de R)
- No soy un programador
- Estoy muy interesado en la creación de modelos/herramientas accesibles

# PROGRAMADOR

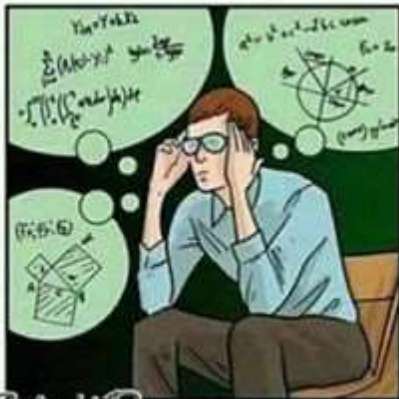
LO QUE LA GENTE  
PIENSA QUE HAGO



LO QUE MIS PADRES  
PIENSAN QUE HAGO



LO QUE YO  
CREO QUE HAGO



LO QUE EN  
REALIDAD HAGO



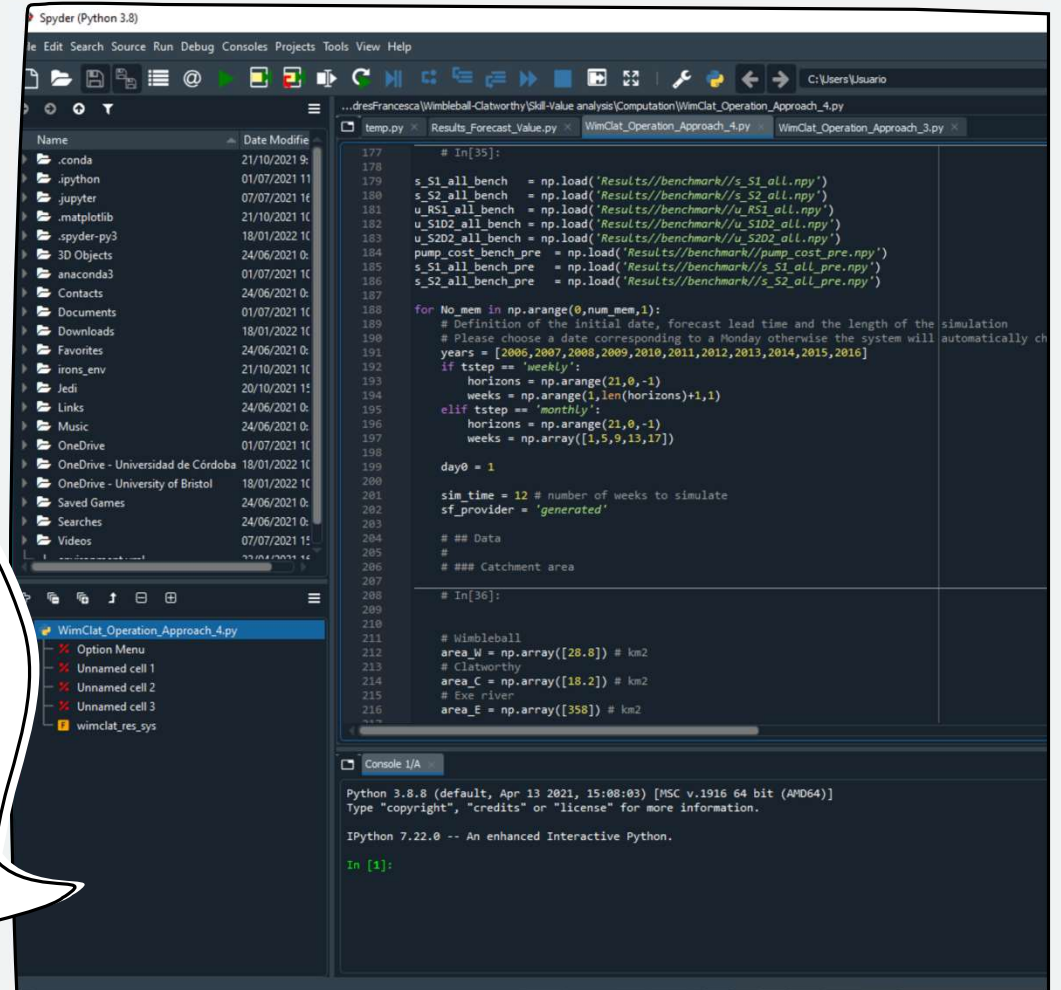
BUENO A LO MEJOR SI  
SOY PROGRAMADOR...



Cuando piensas en programar, ¿imaginas esto?



¿o esto?



The image shows the Spyder Python IDE interface. On the left is a file explorer showing a directory structure with folders like 'conda', 'ipython', 'jupyter', 'matplotlib', 'spyder-py3', '3D Objects', 'anaconda3', 'Contacts', 'Documents', 'Downloads', 'Favorites', 'irons\_env', 'Jedi', 'Links', 'Music', 'OneDrive', 'OneDrive - Universidad de Córdoba', 'OneDrive - University of Bristol', 'Saved Games', 'Searches', and 'Videos'. The main window displays a Python script named 'WimClat\_Operation\_Approach\_4.py'. The script contains several lines of code, including imports, variable assignments, and a loop. The console window at the bottom shows the output of the script, including the Python version (3.8.8) and the IPython version (7.22.0).

```
...desFrancesca(Wimbleball-Clatworthy)\Skill-Value analysis\Computation\WimClat_Operation_Approach_4.py
temp.py Results_Forecast_Value.py WimClat_Operation_Approach_4.py WimClat_Operation_Approach_3.py

177 # In[35]:
178
179 s_S1_all_bench = np.load('Results/benchmark//s_S1_all.npy')
180 s_S2_all_bench = np.load('Results/benchmark//s_S2_all.npy')
181 u_RS1_all_bench = np.load('Results/benchmark//u_RS1_all.npy')
182 u_S102_all_bench = np.load('Results/benchmark//u_S102_all.npy')
183 u_S202_all_bench = np.load('Results/benchmark//u_S202_all.npy')
184 pump_cost_bench_pre = np.load('Results/benchmark//pump_cost_pre.npy')
185 s_S1_all_bench_pre = np.load('Results/benchmark//s_S1_all_pre.npy')
186 s_S2_all_bench_pre = np.load('Results/benchmark//s_S2_all_pre.npy')
187
188 for No_mem in np.arange(0,num_mem,1):
189     # Definition of the initial date, forecast lead time and the length of the simulation
190     # Please choose a date corresponding to a Monday otherwise the system will automatically ch
191     years = [2006,2007,2008,2009,2010,2011,2012,2013,2014,2015,2016]
192     if tstep == 'weekly':
193         horizons = np.arange(21,0,-1)
194         weeks = np.arange(1,len(horizons)+1,1)
195     elif tstep == 'monthly':
196         horizons = np.arange(21,0,-1)
197         weeks = np.array([1,5,9,13,17])
198
199     day0 = 1
200
201     sim_time = 12 # number of weeks to simulate
202     sf_provider = 'generated'
203
204     # ## Data
205     #
206     # ### Catchment area
207
208 # In[36]:
209
210 # Wimbleball
211 area_W = np.array([28.8]) # km2
212 # Clatworthy
213 area_C = np.array([18.2]) # km2
214 # Exe river
215 area_E = np.array([358]) # km2
216
```

Python 3.8.8 (default, Apr 13 2021, 15:08:03) [MSC v.1916 64 bit (AMD64)]  
Type "copyright", "credits" or "license" for more information.  
IPython 7.22.0 -- An enhanced Interactive Python.  
In [1]:

¿Y si programar fuera esto?

## Estimating the model parameters

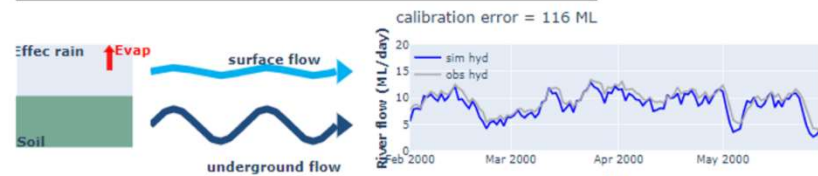
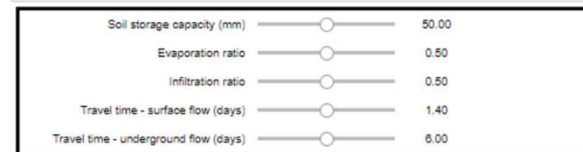
Now imagine we have a time series of measurements of daily rainfall and river flow for a particular catchment. We can use these data to calibrate our model, that is, to find the parameter values that make the model best fit the data. The simplest way to do this is by changing the parameter values one-at-a-time and looking at how this changes the model predictions. Can we fit the predicted river flows to the measured ones?



```
In [1]: from ipywidgets import widgets
from util.hydrological.hydrological_interactive import hydrological_model

Soil_sto, Evap_rate, Inf_rate, Time_surf, Time_under, slider_box_layout, \
fig_sto, fig_flo, fig_hyd, hbox_layout, vbox_layout, param_obs = hydrological_model()

widgets.VBox([widgets.VBox([Soil_sto, Evap_rate, Inf_rate, Time_surf, Time_under], layout=slider_box_layout),
widgets.HBox([fig_sto, fig_flo, fig_hyd], layout=hbox_layout)], layout=vbox_layout)
```



## OBJETIVOS DEL CURSO

- Perder el miedo a programar
- Enseñaros lo útil que es programar y por qué Python es una buena elección
- Aprender algunos conceptos básicos para ya empezar a hacer cosas útiles y prácticas y que os pueden facilitar vuestro trabajo
- Motivaros a que sigáis aprendiendo

# EVALUACIÓN DEL CURSO

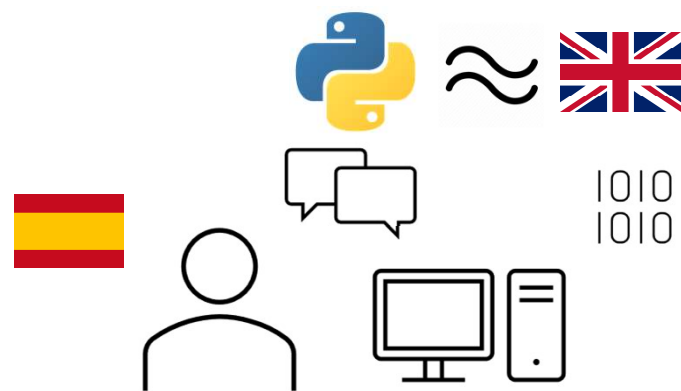
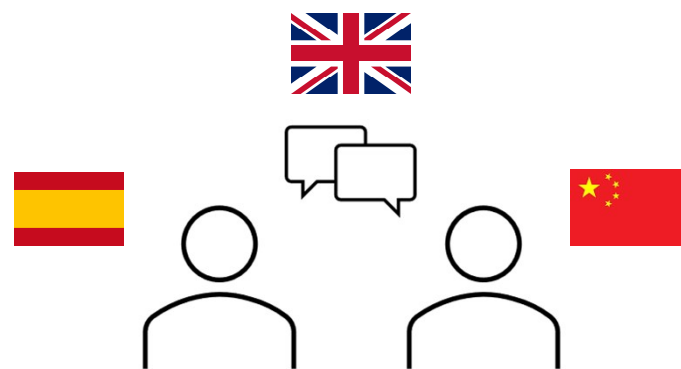
Cuestionarios online	}	50%
Asistencia		
Pequeño proyecto		50%



# ¿QUÉ ES PROGRAMAR?



- Programar es darle instrucciones a tu ordenador para hacer una tarea.
- Escribir las instrucciones en un lenguaje que tu ordenador entienda.
- Nosotros hablamos español pero tu ordenador habla en binario, un lenguaje cuyo alfabeto solo tiene 0 y 1. Entonces, ¿qué podemos hacer para entendernos?
- Para entendernos con nuestro ordenador, necesitamos un traductor y un lenguaje en común, por ejemplo Python.



# TAREA: CEPILLARSE LOS DIENTES

Instrucciones:

1. Coge el cepillo
2. Pon pasta dentífrica en el cepillo
3. Abre la boca
4. Acerca el cepillo a la boca
5. Mueve el cepillo de arriba a abajo
6. Enjuágate la boca




# TAREA: CEPILLARSE LOS DIENTES

Instrucciones para un niño:



1. Coge el cepillo con la mano derecha
2. Pon pasta dentífrica en el cepillo con la mano izquierda
3. Abre la boca como si sonrieras o como si estuvieras muy enfadado
4. Acerca el cepillo a la boca hasta que toque los dientes
5. Mueve el cepillo de arriba a abajo con movimientos cortos que cubran los dientes por completo
6. Enjuágate la boca con agua

# TAREA: CEPILLARSE LOS DIENTES


Instrucciones para un ordenador: 

1. Coge el cepillo con la mano derecha
  - a. Indicar con que mano cogerlo
  - b. Indicar la localización exacta del cepillo
  - c. Definir el cepillo de dientes (por ejemplo material y color)
  - d. Definir la acción "coger": movimientos que tienes que hacer el brazo y la mano para poder coger el cepillo

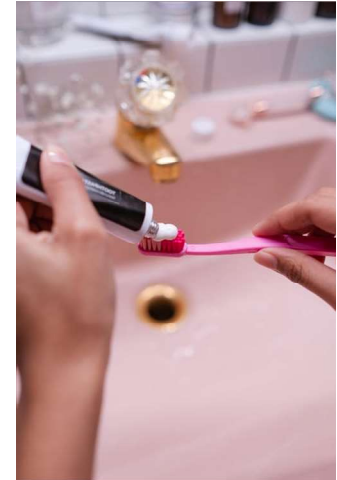




# TAREA: CEPILLARSE LOS DIENTES


Instrucciones para un ordenador: 

2. Pon pasta dentífrica en el cepillo con la mano izquierda
  - a. Indicar con que mano cogerlo
  - b. Indicar la localización exacta de la pasta
  - c. Definir la pasta dentífrica
  - d. Definir la acción "coger" (o utilizar la que hemos utilizado antes)
  - e. Definir la acción "poner": movimientos que tienes que hacer el brazo y la mano para poner la boca de la pasta cerca del cepillo y apretar




... y así para todas las demás sub-tareas

# TAREA: LEER DATOS DE 100 ARCHIVOS BAJADOS DE INTERNET QUE CONTIENEN DATOS DE LLUVIA

Instrucciones para un ordenador: 

1. Baja los archivos de internet
  - a. Indicar la localización donde están los archivos (link)
  - b. Definir que archivos a bajar (fecha y coordenadas del lugar de estudio)
  - c. Crear una carpeta donde vamos a guardar los archivos
  - d. Definir la función para bajar los archivos: acceder al servidor de la pagina web, copiar los archivos a tu disco duro y guardarlos con un determinado nombre cada uno (por ejemplo, indicando la fecha de la predicción de cada archivo)

# TAREA: LEER DATOS DE 100 ARCHIVOS BAJADOS DE INTERNET QUE CONTIENEN DATOS DE LLUVIA

Instrucciones para un ordenador: 

2. Lee los datos que contienen los archivos
  - a. Indicar el tipo de archivo a leer
  - b. Indicar la carpeta donde hemos guardado los archivos en el disco duro
  - c. Definir la función para leer los archivos: capaz de leer el formato de archivo
  - d. Usar dicha función para leer los 100 archivos, extraer (y procesar) los datos que nos interesan y copiarlos a un único archivo mas fácil de leer (por ejemplo, un archivo Excel)

... y así para todas las demás sub-tareas

ES MUY IMPORTANTE QUE EL ORDEN DE LAS TAREAS SEA EL CORRECTO



# ¿POR QUÉ EMPEZAR A PROGRAMAR?

- Programar te ayuda a:
  - automatizar tareas aburridas
  - analizar y visualizar volúmenes grandes de datos
  - crear modelos matemáticos de procesos naturales

# ¿QUE ES PYTHON?



- Es el lenguaje de programación que más rápido está creciendo.

# HISTORIA DE PYTHON

- Creado en 1989 por Guido Van Rossum, un informático de origen holandés cuando trabajaba en un centro de investigación holandés (CWI)
  - Lo creó basándose en otro lenguaje que se llamaba ABC y que era un lenguaje pensado para principiantes por su facilidad de aprendizaje y uso. Y así Python también heredó estas características.

# ¿POR QUÉ SE LLAMA PYTHON?



Pues no es por el tipo de serpiente (pitón) sino en honor a los Monthy Python, el famoso grupo de cómicos británicos.



## CRONOLOGIA DE LAS VERSIONES DE PYTHON

- La versión 1.0, que se publicó en enero de 1994
- La versión 2.0 se publicó en octubre de 2000
- La versión 3.0 se publicó en diciembre de 2008 (y es la que se sigue utilizando en la actualidad)



# FILOSOFÍA DE DESARROLLO DE PYTHON

The Zen of Python, by Tim Peters

Beautiful is better than ugly.

Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.

Sparse is better than dense.

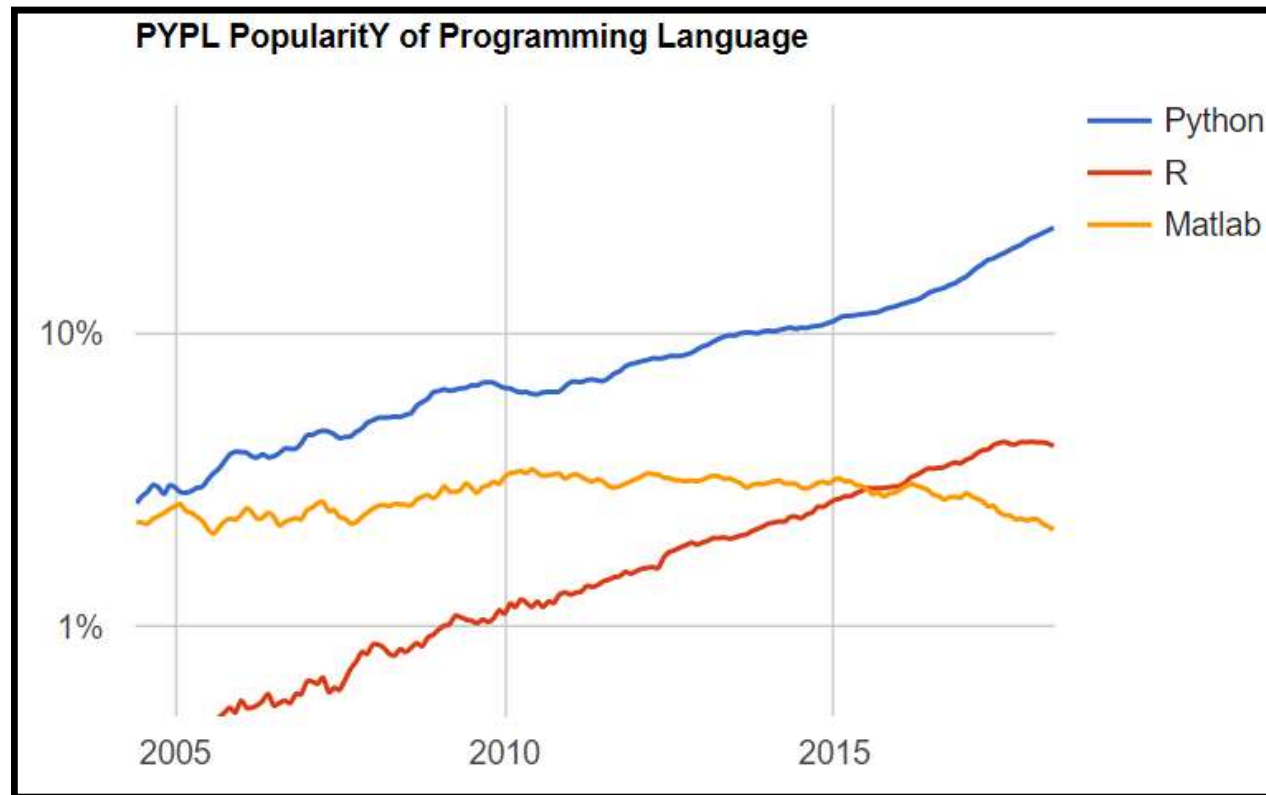
Readability counts.

....

## ¿QUIÉN LO SIGUE MEJORANDO Y ACTUALIZANDO?

- Hasta el año 2018, el desarrollo del lenguaje de programación estaba dirigido personalmente por Van Rossum, pero decidió apartarse
- Desde 2019, son cinco las personas que deciden como evoluciona y se desarrolla Python. Un consejo que se renueva de forma anual.

# EVOLUCIÓN DE LA POPULARIDAD DE PYTHON



¿QUIÉN UTILIZA  
PYTHON?



Lo usan:

- Ingenieros informáticos
  - Matemáticos
  - Contables
- Analistas de datos
  - Científicos

... por qué?

# PYTHON ES MUY VERSÁTIL

Compañías como Amazon, Uber,  
PayPal, Google, Facebook, Instagram,  
y Netflix usan Python

## Se puede usar para:

- Analizar datos (Big Data)
- Visualizar datos
- Creación de webs y apps
- Inteligencia artificial (Machine Learning)
- Automatización de tareas (por ejemplo bajar muchos archivos de una web)
- Crear modelos matemáticos (modelos hidrológicos)



¿PERO POR QUE  
PYTHON Y NO OTRO  
LENGUAJE?



- Es muy versátil
- Puedes resolver problemas complejos con unas pocas funciones (y con poco conocimiento de programación)
- Tiene una comunidad de usuarios inmensa (en Google puedes encontrar la solución a cualquier problema)
- Tiene multitud de herramientas disponibles
- Es un lenguaje relativamente intuitivo y fácil de entender y aprender (se parece al inglés cotidiano)

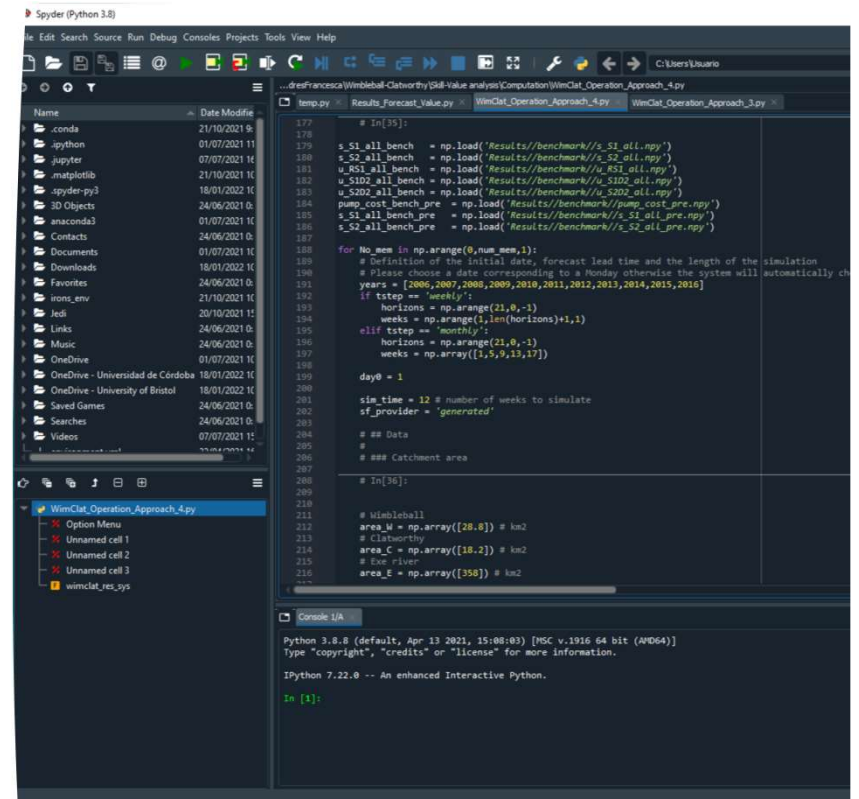
# ¿QUE PROGRAMAS PODEMOS UTILIZAR PARA PROGRAMAR EN PYTHON?

O en otras palabras, ¿que programas podemos usar para traducir Python a lenguaje de 0 y 1?

Cursos (tu conocimiento),  
Google, Manuales...



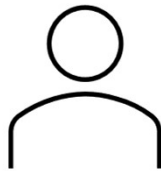
1010  
1010



# ¿QUE PROGRAMAS PODEMOS UTILIZAR PARA PROGRAMAR EN PYTHON?

O en otras palabras, ¿que programas podemos usar para traducir Python a lenguaje de 0 y 1?

Cursos (tu conocimiento),  
Google, Manuales...



1010  
1010

## Estimating the model parameters

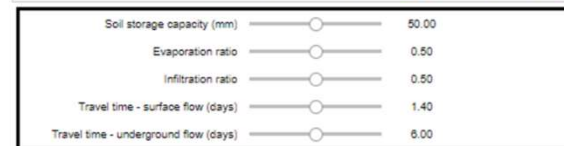
Now imagine we have a time series of measurements of daily rainfall and river flow for a particular catchment. We can use these data to calibrate our model, that is, to find the parameter values that make the model best fit the data. The simplest way to do this is by changing the parameter values one-at-a-time and looking at how this changes the model predictions. Can we fit the predicted river flows to the measured ones?



```
[1]: from ipywidgets import widgets
from util.hydrological.hydrological_interactive import hydrological_model

Soil_sto, Evap_rate, Inf_rate, Time_surf, Time_under, slider_box_layout, \
fig_sto, fig_flo, fig_hyd, hbox_layout, vbox_layout, param_obs = hydrological_model()

widgets.VBox([widgets.VBox([Soil_sto, Evap_rate, Inf_rate, Time_surf, Time_under], layout=slider_box_layout),
widgets.HBox([fig_sto, fig_flo, fig_hyd], layout=hbox_layout)], layout = vbox_layout)
```



# JUPYTER NOTEBOOKS

Entorno de programación.

Documento en el que podemos combinar:

- Texto: explicación del código y de la metodología o ecuaciones aplicadas.
- Imágenes o diagramas
- Código: el cual podemos ver y ejecutar
- Elementos interactivos: facilitar la interacción del usuario con el código
- Gráficas: visualizar los resultados después de ejecutar el código

Texto

## Estimating the model parameters

Now imagine we have a time series of measurements of daily rainfall and river flow for a particular catchment. We can use these data to calibrate our model, that is, to find the parameter values that make the model best fit the data. The simplest way to do this is by changing the parameter values one-at-a-time and looking at how this changes the model predictions. Can we fit the predicted river flows to the measured ones?

Imágenes



```
In [1]: from ipywidgets import widgets
from util.hydrological.hydrological_interactive import hydrological_model

Soil_sto, Evap_rate, Inf_rate, Time_surf, Time_under, slider_box_layout, \
fig_sto, fig_flo, fig_hyd, hbox_layout, vbox_layout, param_obs = hydrological_model()

widgets.VBox([widgets.VBox([Soil_sto, Evap_rate, Inf_rate, Time_surf, Time_under], layout=slider_box_layout),
widgets.HBox([fig_sto, fig_flo, fig_hyd], layout=hbox_layout)], layout = vbox_layout)
```

Código

Soil storage capacity (mm)	<input type="text" value="50.00"/>	50.00
Evaporation ratio	<input type="text" value="0.50"/>	0.50
Infiltration ratio	<input type="text" value="0.50"/>	0.50
Travel time - surface flow (days)	<input type="text" value="1.40"/>	1.40
Travel time - underground flow (days)	<input type="text" value="6.00"/>	6.00

Elementos  
interactivos

Gráficas



¿Alguna pregunta?

# COMO INSTALAR PYTHON Y JUPYTER NOTEBOOK EN NUESTRO ORDENADOR

Para ello vamos a bajar e instalar el paquete de programas gratuito Anaconda (el cual incluye Python y Jupyter Notebook entre otras aplicaciones como Spyder)

1. En este link bajar "Anaconda Individual Edition":

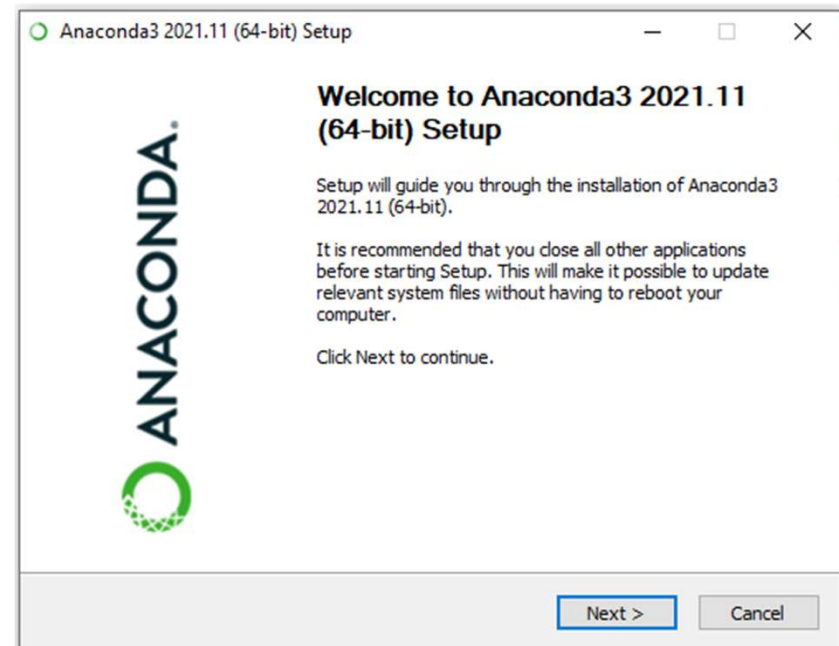
<https://www.anaconda.com/products/individual>

Encontraras versiones para Windows y Mac

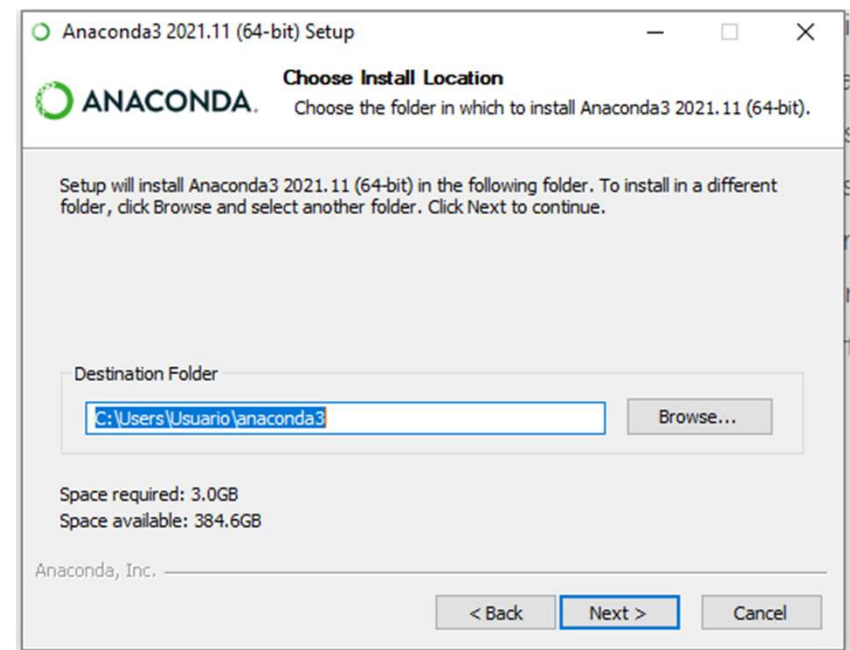
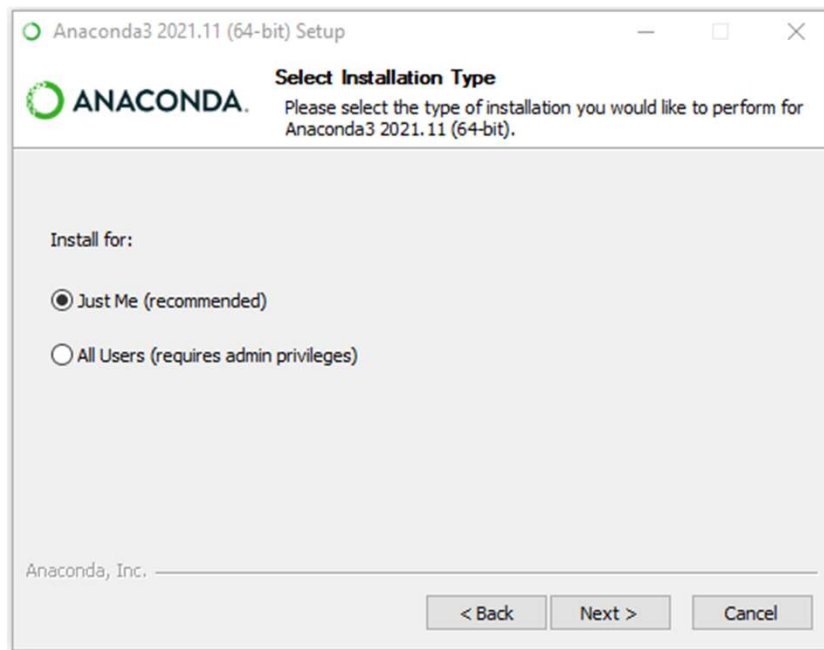


# COMO INSTALAR PYTHON Y JUPYTER NOTEBOOK EN NUESTRO ORDENADOR

2. Seguir las instrucciones de instalación

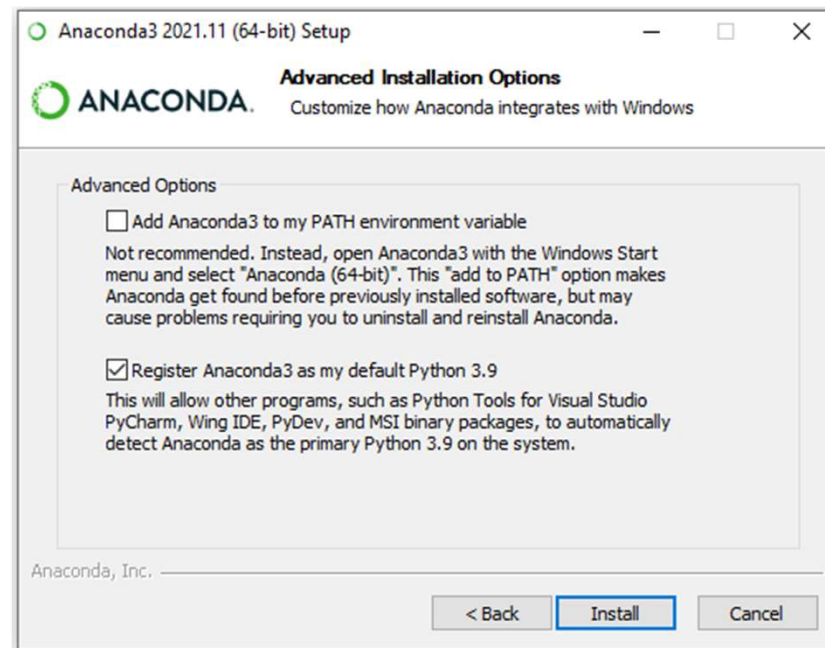


# COMO INSTALAR PYTHON Y JUPYTER NOTEBOOK EN NUESTRO ORDENADOR



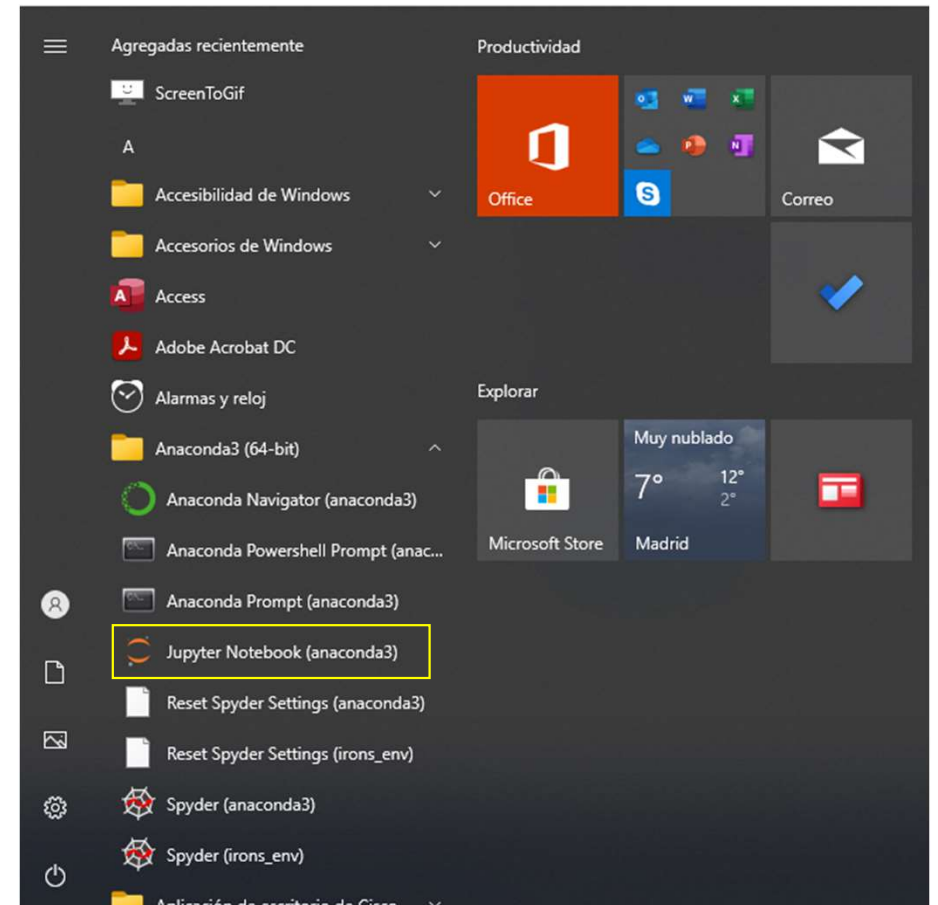


# COMO INSTALAR PYTHON Y JUPYTER NOTEBOOK EN NUESTRO ORDENADOR



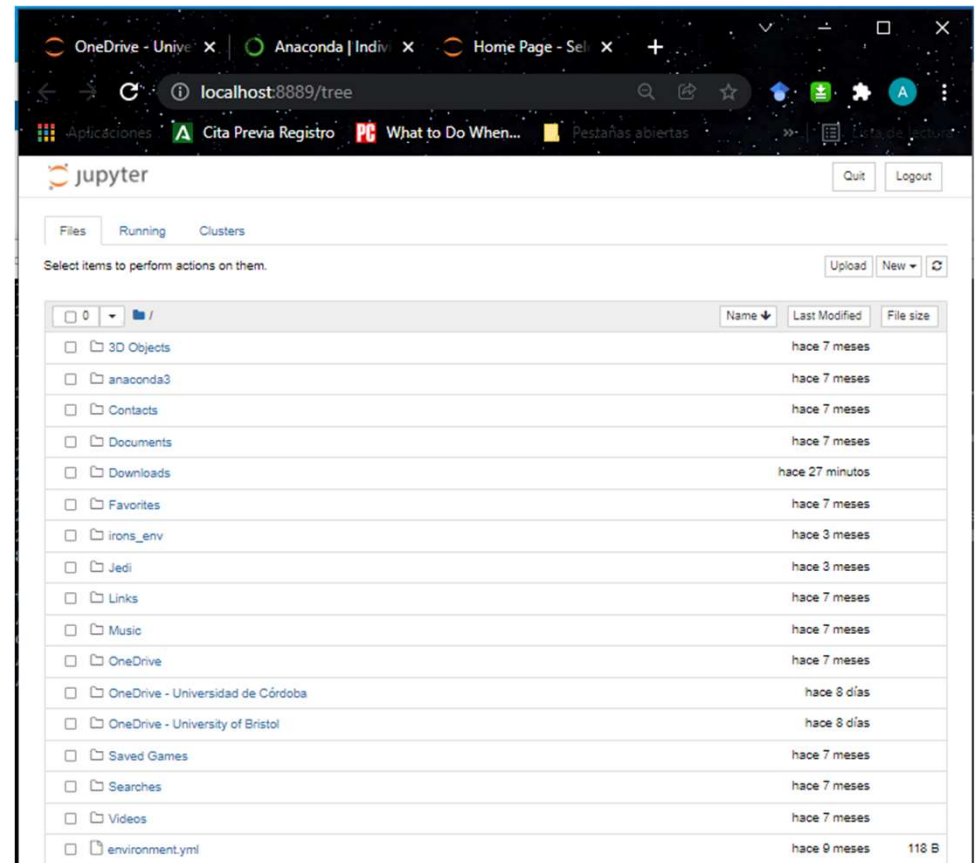
# UNA VEZ INSTALADO COMO EJECUTAMOS JUPYTER NOTEBOOK

1. Abrimos el menú de Windows
2. Buscamos Anaconda3 y hacemos click con el raton
3. Ahora buscamos Jupyter Notebook y hacemos click



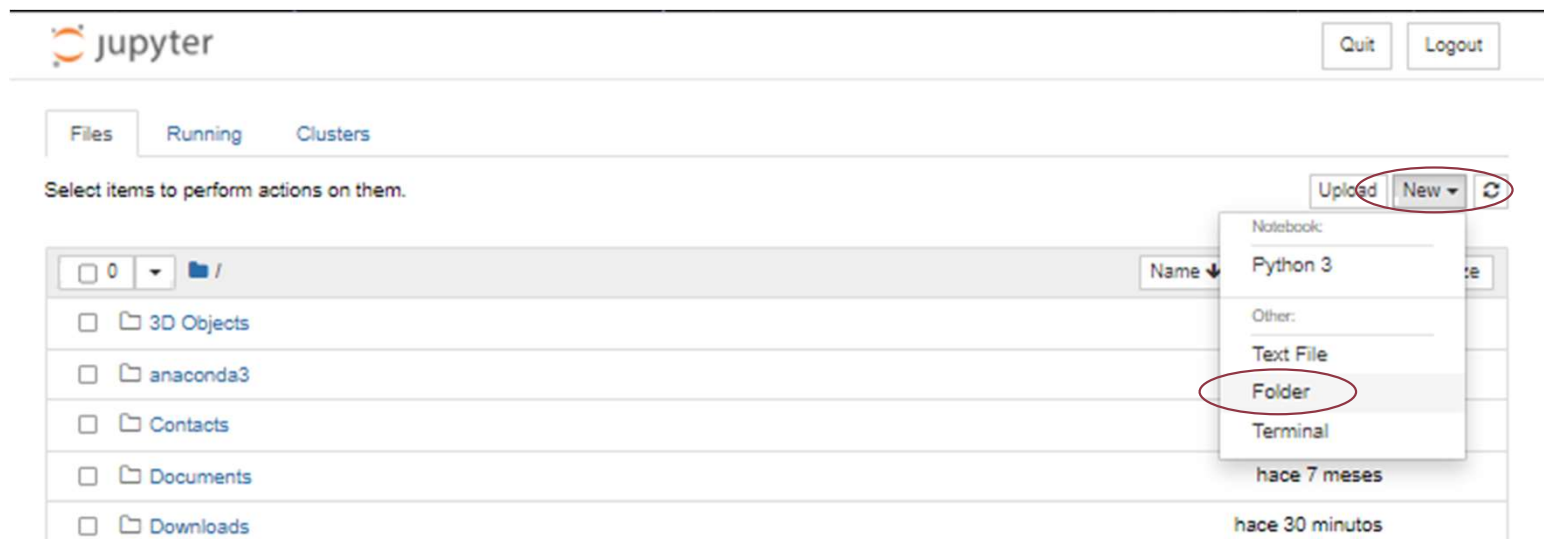
# QUE DEBÉIS VER AL EJECUTAR JUPYTER NOTEBOOK

Se abrirá automáticamente vuestro web browser predeterminado (por ej., Chrome) y aparcerá una pestaña como esta.



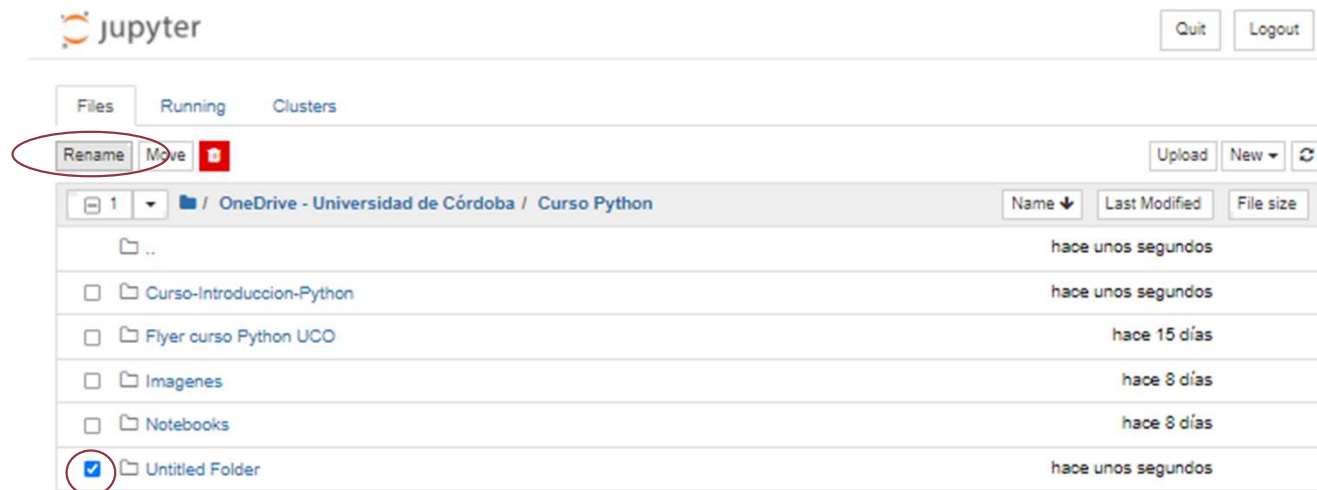
# CREAR CARPETA PARA EL CURSO

A continuación podéis crear una carpeta nueva en vuestro disco duro donde vais a trabajar durante el curso y donde se van a guardar los Notebooks del curso. Para ello primero haz click in "New" y después en "Folder"



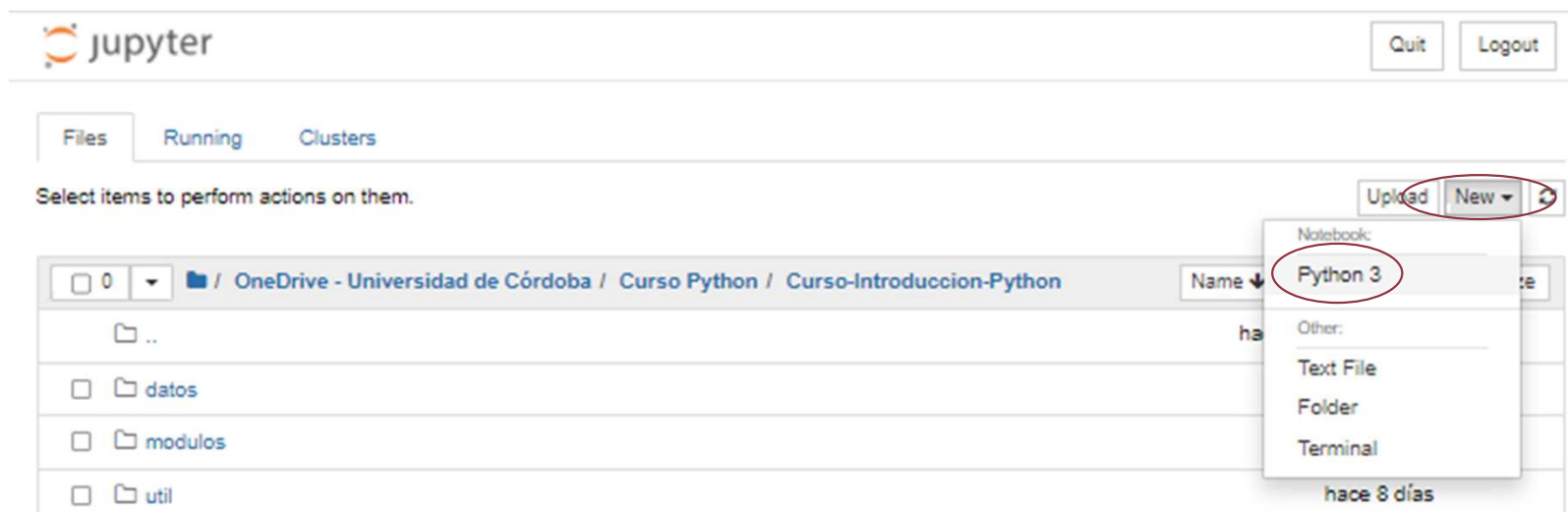
# CREAR CARPETA PARA EL CURSO

Se creará una carpeta con el nombre "Untitled Folder" la cual podemos renombrar como queramos, por ejemplo "Curso-Introducción-Python". Para ello primero seleccionamos la carpeta y después hacemos click en "Rename"

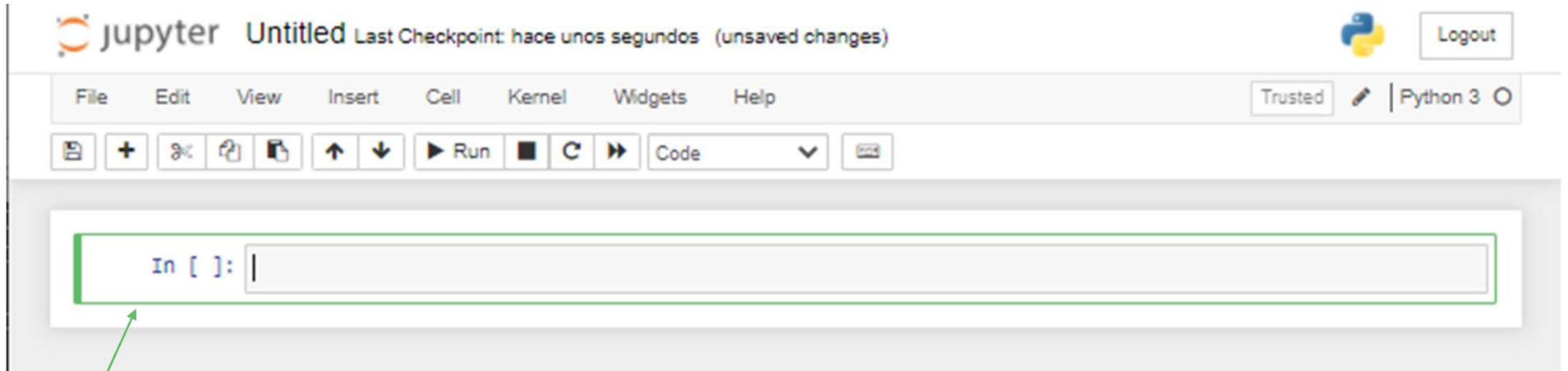


# CREAR UN NUEVO NOTEBOOK

Una vez dentro de la carpeta que habéis creado, por ejemplo "Curso-Introducción\_Python", vamos a crear un nuevo Notebook en blanco donde podemos empezar a trabajar.



# ¡YA PODEMOS EMPEZAR A PROGRAMAR!



Celda

## SI NO HAS PODIDO INSTALAR ANACONDA...

En este link podrás abrir un Notebook en blanco y empezar a programar, solo necesitas tu explorador de internet (por ejemplo, Chrome):

<https://mybinder.org/v2/gh/AndresPenuela/Curso-Introduccion-Python/HEAD?urlpath=tree/Untitled.ipynb>