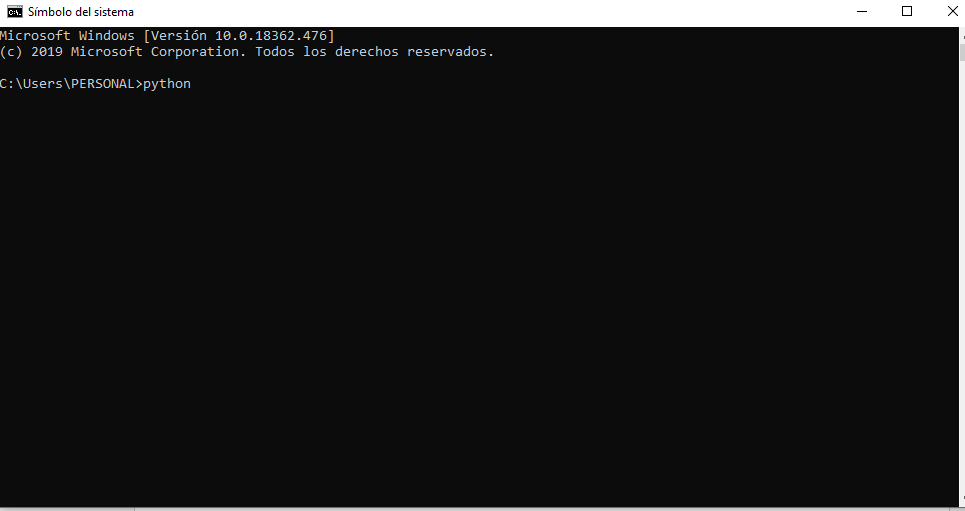
Primero, como no es un ejecutable, es necesario tener python instalado y las librerías instaladas.

NOTA: Se programó el pipeline (el archivo simulation.py) para un sistema operativo Windows por lo que el simulador no corre en otro sistema operativo.

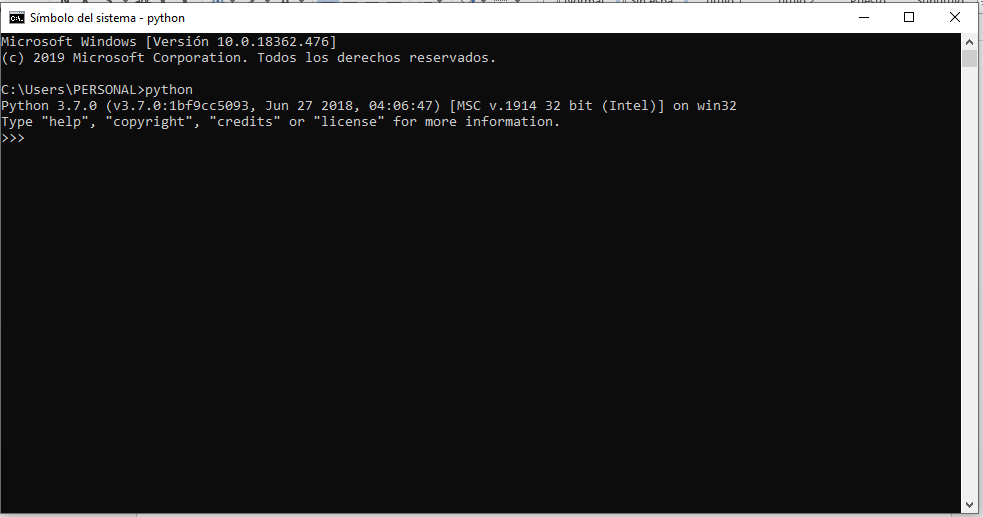
Python se debe instalar de:

<https://www.python.org/downloads/>

Una vez se tenga eso es necesario revisar que python se pueda ejecutar desde la línea de comandos. Esto se hace abriendo la línea de comandos y escribiendo el comando Python



Debe aparecer lo siguiente al darle enter

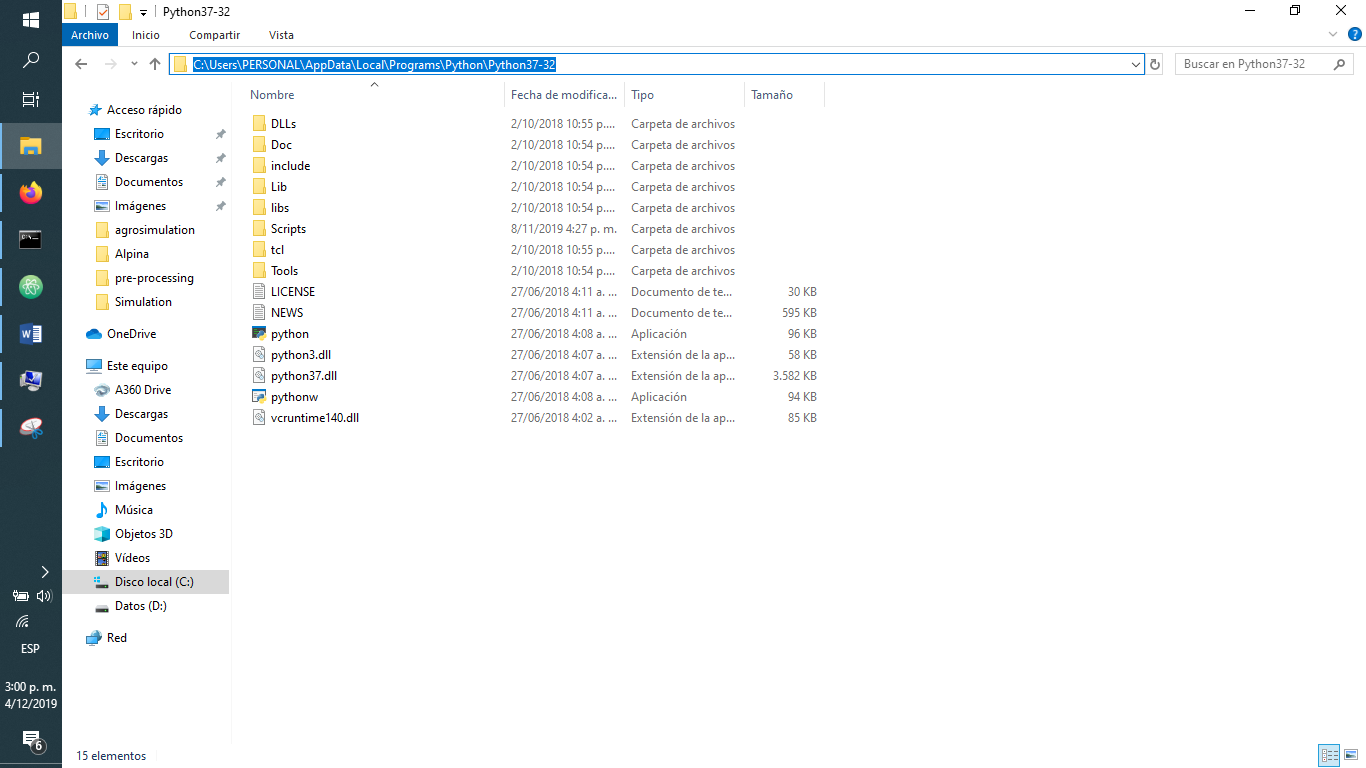


Si aparece que el comando no es reconocido, se debe poner a Python como una variable de entorno.

Si es así, les recomiendo que sigan las instrucciones del siguiente vídeo

<https://www.youtube.com/watch?v=hX4yh8eLqlE>

Si no encuentran la ruta que indica en el vídeo, les recomiendo que la pongan directamente, lo único que cambia es el nombre del usuario, en mi caso es PERSONAL



C:\Users\”NOMBRE DE USUARIO”\AppData\Local\Programs\Python\Python37-32

C:\Users\”NOMBRE DE USUARIO” \AppData\Local\Programs\Python\Python37-32\Scripts

Una vez que tengan esas dos rutas en el path como indica el vídeo, vuelvan a intentar los pasos de la línea de comandos, hasta que les dé el resultado indicado.

Una vez tengan eso, ejecutar los siguientes comandos:

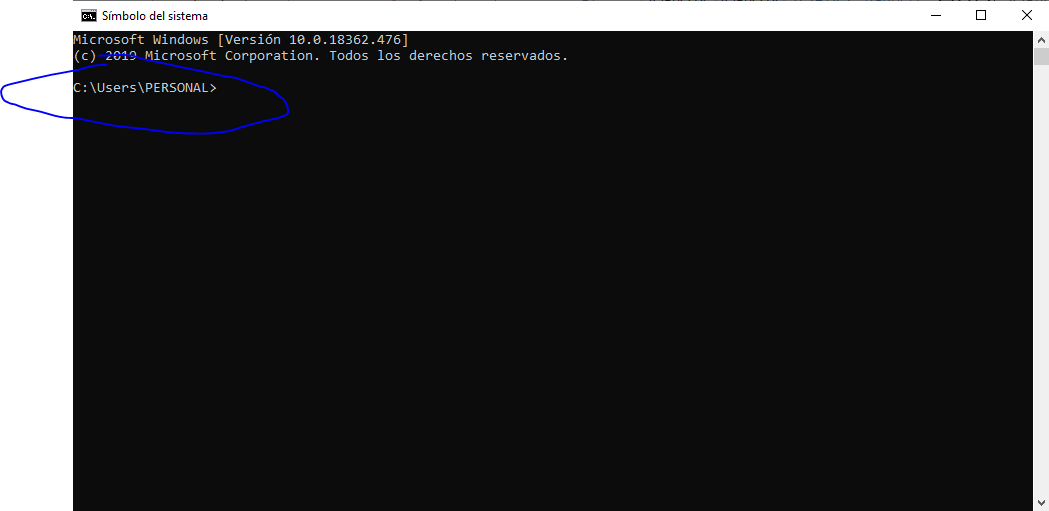
python -m pip install --upgrade pip

pip install numpy pandas matplotlib seaborn scipy

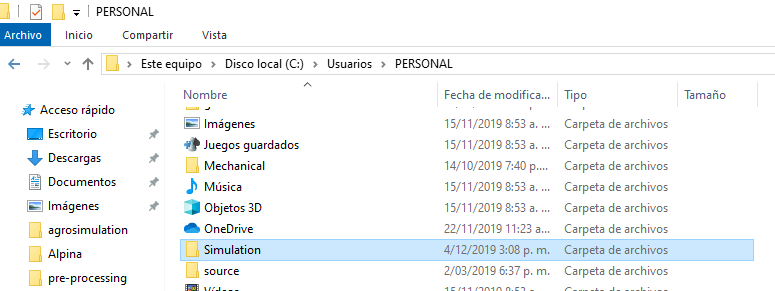
**EJECUCIÓN DEL SIMULADOR**

Con esto tienen las librerias instaladas. Ahora si pueden ejecutar el simulador.

Antes de hacerlo deben ir a través del command line al directorio en el que está el archivo

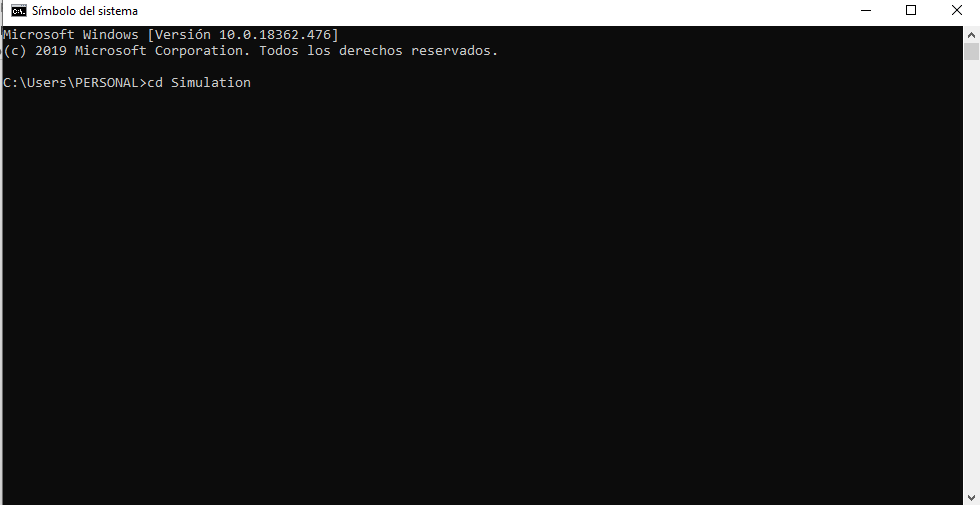


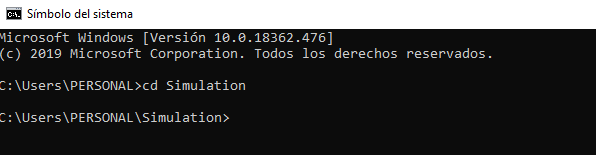
Cuando se abre el cmd aparece generalmente en esa dirección. Una forma de ahorrarse el desplazamiento por el cmd es guardando el archivo en esa carpeta



Una vez ahí se debe ejecutar el comando en el cmd

Cd Simulation



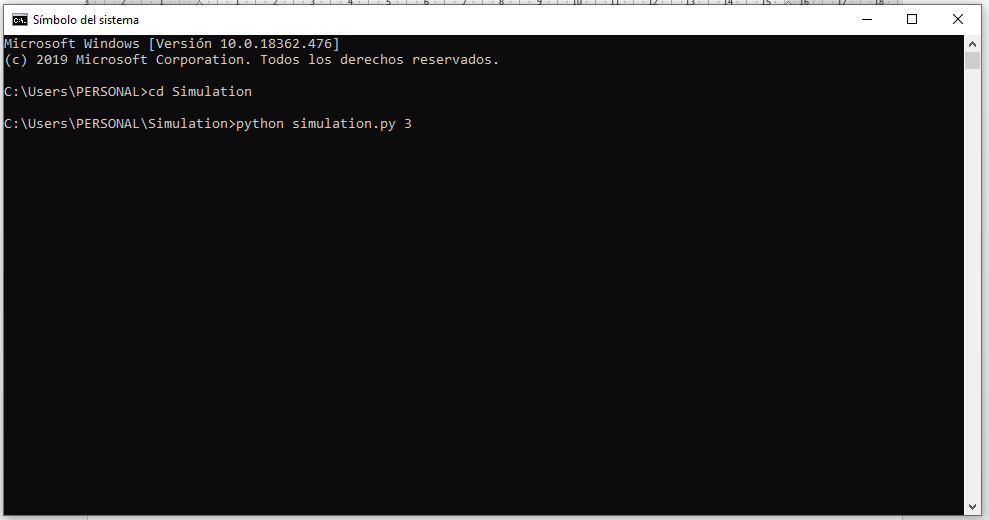


Estamos dentro de la carpeta.

Ahora, basta con poner el siguiente comando para ejecutar el simulador

Python simulation.py 4

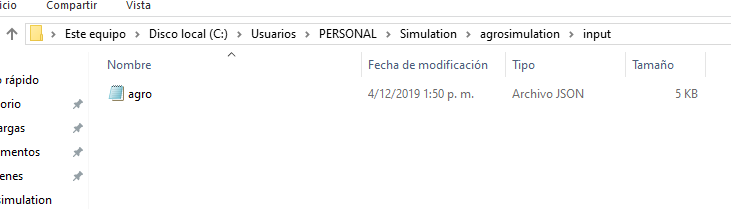
Este último número representa el número de iteraciones que realiza el simulador



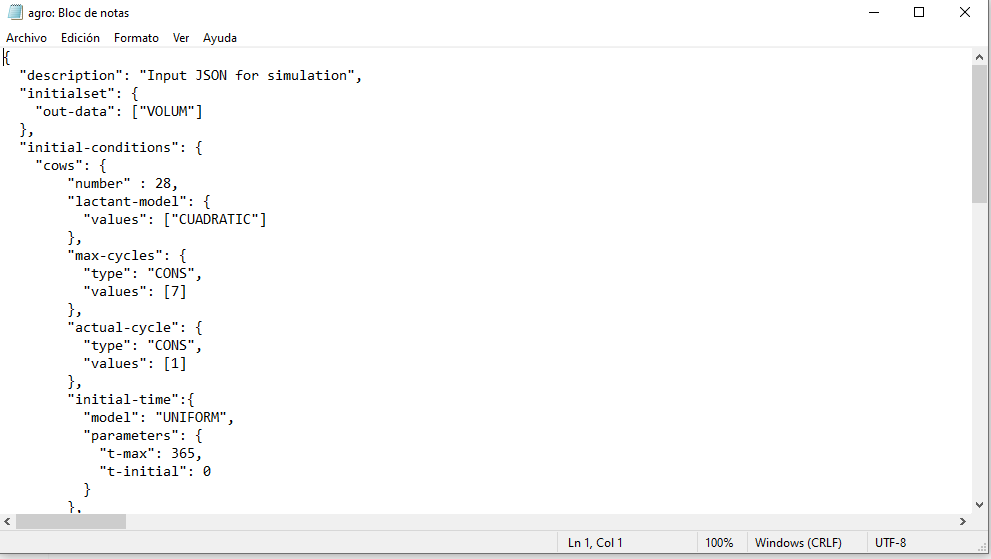
Una vez le den enter, tendrán el simulador funcionando.

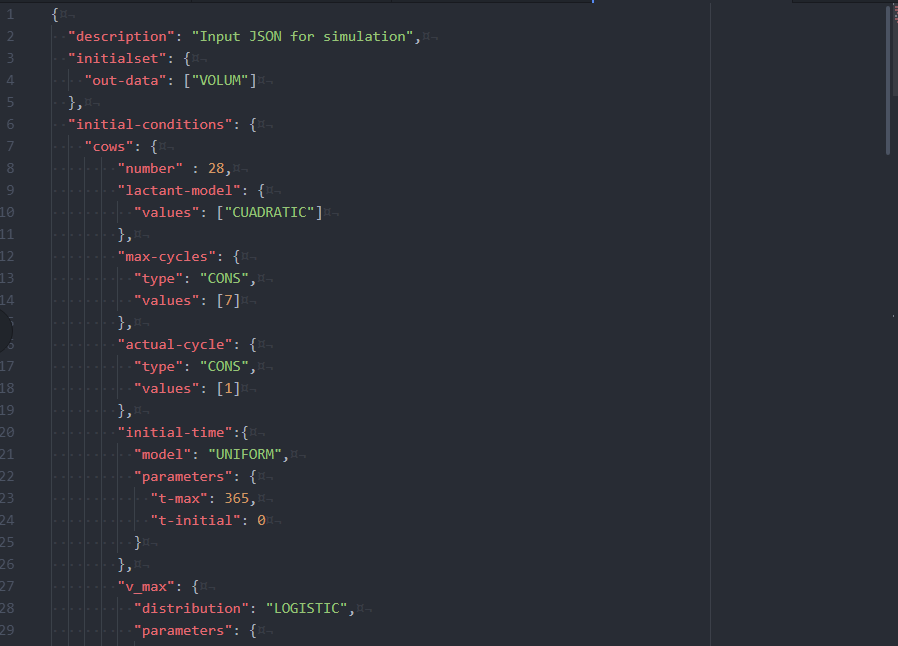
INGRESO DE DATOS

Éste se hace con un archivo formato .JSON, buscar dentro de la carpeta del simulador, la carpeta agrosimulation y dentro de esta abrir la carpeta input



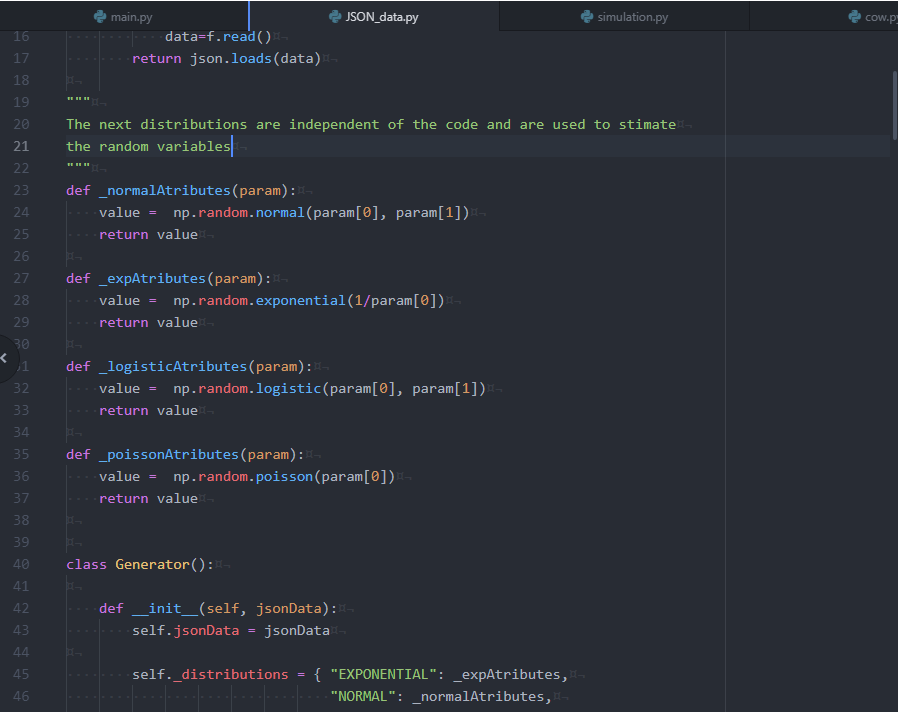
Ese archivo contiene las entradas. Pueden abrirlo con cualquier editor de texto. Se puede abrir y editar con el bloc de notas. Sólo que es incómodo, porque si falta una coma, o se borra algún símbolo del bloc y no sabemos qué es, no va a funcionar el simulador. Yo lo abro con el editor de texto atom, y ese mismo archivo se ve en la siguiente imagen. Existen otros editores de texto muy livianos y fáciles de instalar, con los que se puede editar el JSON cómodamente. Recomiendo, sublime text, visual studio code, atom, y si tienen otros gustos, no hay problema, lo importante es que se sientan cómodos editando el JSON





Dentro del JSON se puede ver los nombres de las variables aleatorias del sistema, la distribución que usan y los parámetros que tienen. Tener cuidado con poner distribuciones que no estén programadas.

**¿DÓNDE Y CÓMO PROGRAMO NUEVAS DISTRIBUCIONES?**

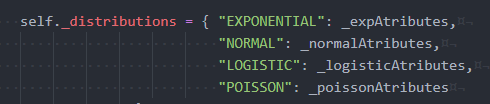




Sólo es necesario que modifiquen esa parte para agregar distribuciones. Ese módulo está en agrosimulation y el archivo se llama JSON\_data.py

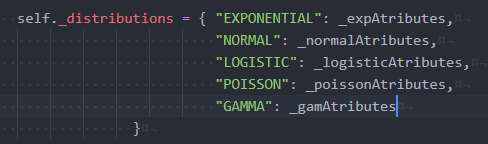
Eso responde el dónde.

Para agregar nuevas distribuciones, agreguen el nombre de la distribución en inglés (estas son sólo convenciones para que el código sea fácilmente editable en próximas ocasiones), y el nombre de la función que la representa (esta función se programa después).



Por ejemplo si quiero agregar la distribución gamma, haré lo siguiente:

Creo un nuevo elemento llamado “GAMMA” (ponerlo en mayúscula), y el nombre de la función \_gamAtributes, quedaría así:

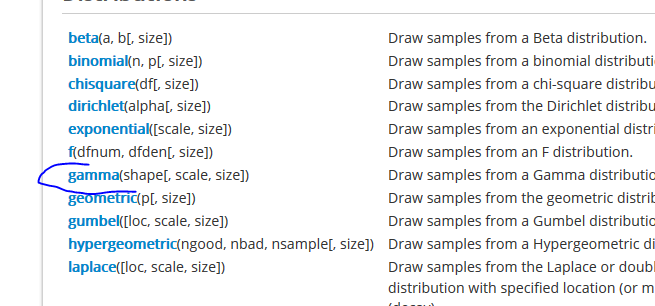


Esta función aún no existe, y no está programada, pero esto se hará en breves. Primero, mirar la convención de la función, barra al piso ‘\_’, luego un distintivo de la función ‘gam’ y luego la palabra ‘Atributes’, les pido encarecidamente que la mantengan.

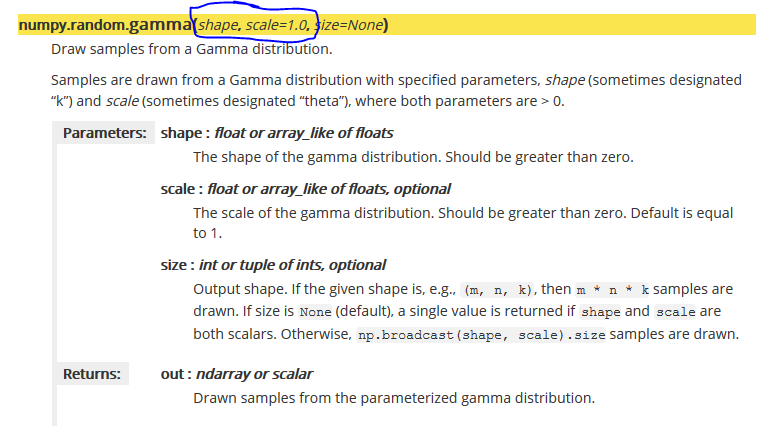
Una vez hecho esto, vamos a agregar los parámetros. Revisar la librería para observar cuáles son los que se permiten en Python. En la siguiente ruta está la página

<https://docs.scipy.org/doc/numpy-1.15.0/reference/routines.random.html>

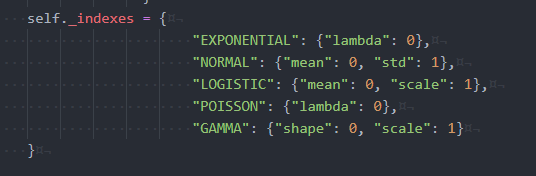
Busquen la gamma



Y háganle click



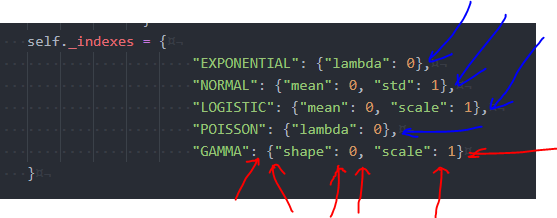
Observen que tiene dos parámetros, shape and scale. Estos los vamos agregar en la siguiente parte:



Agreguenlos de la siguiente manera:

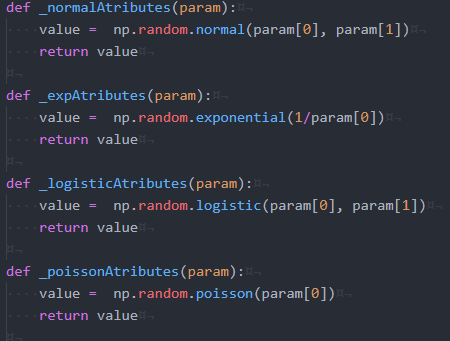
Los parámetros incluirlos en minúsculas, el número que acompaña al parámetro NO es el valor del parámetro, es el índice del parámetro y que será necesario tenerlo en cuenta para la programación de la función.

OJO con lo siguiente



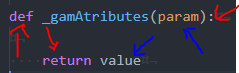
Los primeros elementos tienen comas, y el último no. Los dos puntos son para generar una relación de diccionario donde el elemento de la izquierda es una llave y el de la derecha es el valor. y la coma intermedia sirve para separar parámetros. Eso podría generarle errores. En caso de que no les funcione al correr, deberían revisar eso.

Una vez definido parámetros y el nombre de la función, se debe programar la función:

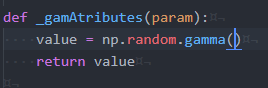


Esta es la estructura general.

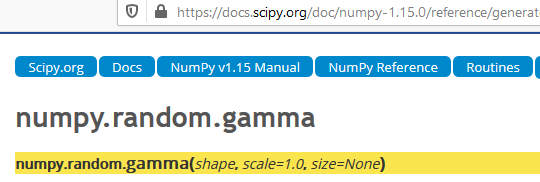
Iré construyendo la ecuación por pasos



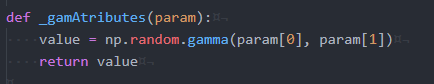
Lo que señalo en rojo es obligatorio, y lo que señalo con azul es la estructura del código que estoy manejando. El nombre de la función es la misma que se definió en el diccionario inicial de las distribuciones



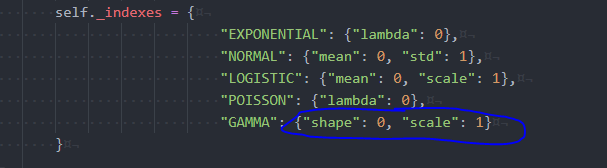
El valor que retornaré es aleatorio, por lo que tomaré el valor de ese módulo de la librería de Python, ese módulo es np.random.gamma() , ya les pasé el link.



Al agregar los parámetros la función quedaría como:



El param vendrá del diccionario que tiene los índices



De esta manera, sólo deben asegurarse de poner dentro de la función la parama con el índice que programaron en \_indexes. Param[0] hace referencia a “shape” y param[1] hace referencia a “scale”. Tener cuidado de que ese sea el orden en que ingresan a la función



Primero entra shape y luego scale