Glosario

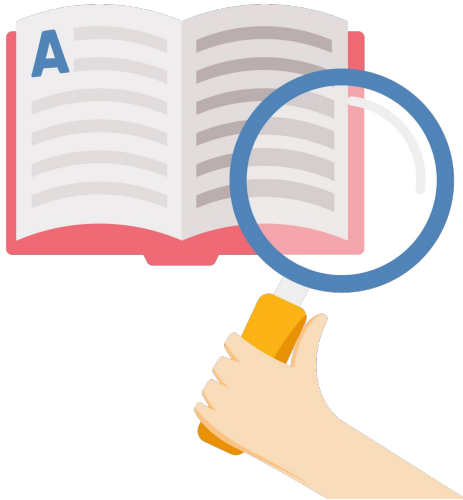
Python para Análisis de datos

- **Jupyter Notebook:** entorno interactivo para trabajar con lenguajes de programación. Es una aplicación web, por lo que se visualiza en cualquier navegador. Está pensado para articular código ejecutable, gráficos, texto, tablas, fórmulas, etc. en un solo documento.

- **Dato**: representación simbólica de algún aspecto de la realidad. Debe ser adecuado para su comunicación e interpretación.

- **Operadores aritméticos**: (+, -, *, /, //, **, %) operadores binarios que permiten realizar operaciones entre números, aunque algunos también funcionan para colecciones, como + y *.

- **Operadores de comparación**: los operadores de comparación por valor son ==, !=, >, >=, <, <=, comparan los valores respectivos de los operandos que no necesitan ser del mismo tipo de dato.



- **Operadores de pertenencia:** para chequear pertenencia tenemos los operadores in y not in.

- **Operadores de identidad**: (is e is not) permiten evaluar si los operandos son el mismo objeto en la memoria, lo que internamente se determina usando la función id().

- **Operadores lógicos**: operan sobre booleanos y devuelven un booleano. Se trata de los operadores de conjunción, disyunción y negación, cuyos operadores son and, or y not.

- **Colecciones**: Python viene con una variedad de colecciones para agrupar datos en estructuras diferentes. Tenemos colecciones ordenadas y no ordenadas y colecciones mutables e inmutables.

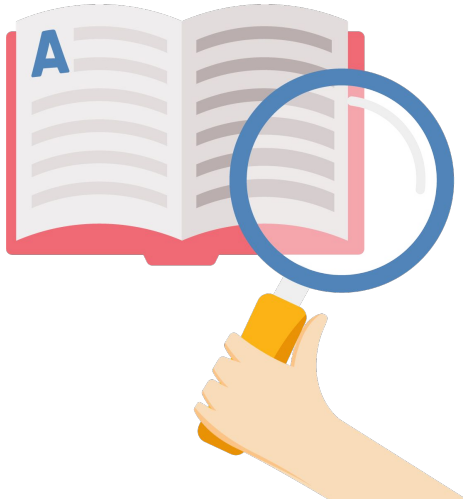
- **Estructuras de control**: permiten controlar el flujo de ejecución de un programa, que de otro modo es de arriba hacia abajo. Las más básicas son el condicional y los bucles, aunque funciones y clases también son estructuras de control.

- **Condicional:** permite decidir si se ejecuta un bloque de código o no en base a una condición. Se utilizan las palabras if, elif y else en distintas combinaciones. En Python no existe la estructura switch/case, pero se puede recrear con if/elif.

- **Listas por comprensión:** permiten crear listas de una forma muy concisa. Consiste en especificar los elementos que va a contener la lista en la propia definición de la lista.

- **Funciones**: bloques de código que se pueden agrupar bajo un nombre. Esto permite que se pueda ejecutar ese código (por más complejo que sea) llamando al nombre de la función.

- **Definición**: para definir una función se utiliza la palabra reservada def. Las reglas para nombrar funciones son las mismas que para las variables. Entre paréntesis va la lista de parámetros y luego de los dos puntos va el bloque de código que constituye el cuerpo de la función.



- **Argumentos:** a una función se le puede pasar información en forma de argumentos. Estos se definen entre los paréntesis que siguen al nombre de la función en la definición y pueden ser tantos como se necesiten.

- **Funciones de orden superior**: son aquellas que toman a otras funciones como parámetros o devuelven a una función.

- **Funciones anónimas o lambda**: no tienen nombre y su cuerpo está limitado a una única expresión.

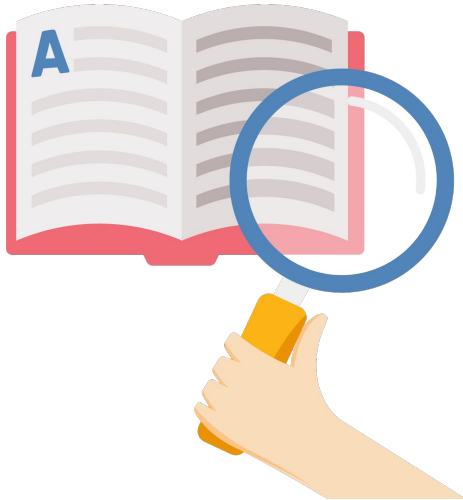
- **Numpy (de Numerical Python)**: principal librería de cálculo numérico de Python y el ecosistema de librerías científicas del lenguaje (como Pandas y Matplotlib) recurre a Numpy para realizar sus operaciones.

- **Array:** estructura que permite acelerar las operaciones matemáticas en Python sin la necesidad de recurrir a bucles para operar con todos sus elementos, lo que se llama vectorización.

- **Indexing:** cualquier operación que seleccione elementos de un array a través de corchetes ([]).

- **Pandas**: es un módulo muy popular para la manipulación y análisis de datos. Está hecho sobre Numpy y provee estructuras de datos diseñadas para trabajar con datos con tablas. A diferencia de un array de numpy, Pandas permite trabajar con distintos tipos de datos para cada columna asemejándose más a lo que se usa en bases de datos relacionales.

- **Serie**: estructura de datos unidimensional donde cada dato tiene además una etiqueta. Es una estructura mutable a la que no sólo se le puede cambiar sus elementos (como en un array de numpy) si no que además se le pueden agregar y quitar elementos (a diferencia de un array).



- **Visualización de datos**: representación de los datos a través de gráficos de distinto tipo. Es un tipo de representación mucho más fácil de entender para una persona que otras formas de representar información, como las tablas, más aún cuando la cantidad de datos es grande.

- **Matplotlib**: módulo de bajo nivel para la visualización de datos. Está hecho sobre Numpy, por lo que es ideal para hacer gráficos a partir de datos alojados en arrays. Por esta razón se integra perfectamente al ecosistema de módulos de Python.

¡Ya estás listo!