Python es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning (ML). Los desarrolladores utilizan Python porque es eficiente y fácil de aprender, además de que se puede ejecutar en muchas plataformas diferentes. El software Python se puede descargar gratis, se integra bien a todos los tipos de sistemas y aumenta la velocidad del desarrollo.

* Tipos de datos

Un tipo de dato se refiere a la clasificación de los valores que pueden ser almacenados en una variable. Cada tipo de dato tiene características y propiedades específicas que lo hacen adecuado para ciertos usos y operaciones.

Los tipos de datos también se utilizan para determinar cómo se almacenan los valores en la memoria de la computadora y cómo se pueden manipular y operar los valores. Por ejemplo, los números enteros se almacenan como secuencias de bits en la memoria, y se pueden realizar operaciones aritméticas como suma, resta, multiplicación y división.

1. Numéricos
2. De texto
3. Booleanos
4. De secuencias
5. De mapeo
6. Conjuntos
7. De bytes

Tipos de datos numéricos

int

Los datos tipo int (abreviatura de «entero»), en Python, son aquellos que representan números enteros. Estos son un tipo de número que no tiene parte decimal, es decir, son completos sin fracciones. En Python, los datos tipo int se representan simplemente escribiendo un número sin parte decimal.

float

Los datos tipo float (abreviatura de «coma flotante»), en Python, son aquellos que representan números con decimales. En otras palabras, los datos tipo float son números reales que pueden tener una parte entera y una parte decimal. En Python, los datos tipo float se representan escribiendo un número con un punto decimal.

complex

Los datos tipo complex (abreviatura de «números complejos»), en Python, son aquellos que representan números que tienen una parte real y una parte imaginaria. En matemáticas, los números complejos se definen como la suma de una parte real y una parte imaginaria multiplicada por la unidad imaginaria (que se denota como «i» o «j»).

Los datos tipo complex se representan escribiendo un número real seguido por una «j» para indicar la parte imaginaria.

Tipos de texto

str

Los datos tipo str (abreviatura de «cadena de caracteres» o «string»), en Python, son aquellos que representan una secuencia de caracteres. En otras palabras, los datos tipo string son texto que se puede representar mediante una serie de caracteres (letras, números, símbolos, etcétera). Los datos tipo string se representan entre comillas simples ('') o dobles ("").

Los siguientes son ejemplos de datos tipo str en Python:

"Hola mundo", 'Python es genial', "1234".

Tipos booleanos

bool

Los datos tipo bool (abreviatura de «booleanos»), en Python, son aquellos que representan valores de verdad. En otras palabras, son valores verdaderos o falsos. En Python, los datos tipo bool se representan como True o False (con la primera letra en mayúscula).

Tipos de secuencias

list

Los datos tipo list (abreviatura de «listas»), en Python, son aquellos que representan una colección ordenada de elementos. En otras palabras, las listas son una estructura de datos que permite almacenar múltiples valores en un solo objeto y acceder a ellos por su posición en la lista.

tuple

Los datos tipo tuple (abreviatura de «tuplas»), en Python, son similares a las listas, pero a diferencia de las listas, son inmutables. En otras palabras, las tuplas son una estructura de datos que permite almacenar múltiples valores en un solo objeto y acceder a ellos por su posición en la tupla, pero una vez creada la tupla, los elementos no pueden modificarse.

Los datos tipo tupla se representan mediante paréntesis «( )» que contienen los elementos de la tupla separados por comas. Los elementos pueden ser de cualquier tipo de datos, incluyendo otros objetos, como otras tuplas.

range

Los datos tipo range, en Python, representan una secuencia inmutable de números enteros. Un objeto de tipo range se utiliza comúnmente para generar una secuencia de números enteros para su uso en un bucle for.

La función range() devuelve un objeto de tipo range. El objeto range toma tres argumentos: el valor inicial, el valor final (no incluido en la secuencia) y el tamaño del paso. Por defecto, el valor inicial es 0 y el tamaño del paso es 1.

Tipos de mapeo

dict

Los datos tipo dict (abreviatura de «diccionario»), en Python, son una estructura de datos que permite almacenar un conjunto de datos como pares clave-valor, donde cada valor es accesible a través de una clave única. En otras palabras, los diccionarios permiten asociar valores con claves.

Los datos tipo dict se representan mediante llaves «{ }» que contienen una serie de pares clave-valor separados por comas y cada par se separa por dos puntos «:». Las claves en un diccionario son únicas y pueden ser de cualquier tipo inmutable, como una cadena, un número entero o una tupla. Los valores también pueden ser de cualquier tipo de datos.

Tipos de conjuntos

set

Los datos tipo set, en Python, son una colección de elementos únicos e inmutables; es decir, no debe haber duplicados en un conjunto y no se pueden cambiar después de ser creados. Un conjunto se crea utilizando llaves «{ }» o la función set() y los elementos se separan por comas.

frozenset

Los datos tipo frozenset son similares a los conjuntos (sets), pero son inmutables; es decir, una vez creados no se pueden modificar. Se crean utilizando la función frozenset() y se usan para almacenar elementos únicos, como los conjuntos.

A diferencia de los conjuntos, los datos tipo frozenset se pueden utilizar como elementos de diccionarios, ya que son inmutables y, por lo tanto, hashables. Además, los datos tipo frozenset es factible usarlos como elementos de otros conjuntos, ya que también son inmutables.

Tipos de bytes

bytes

Los datos tipo bytes, en Python, son una secuencia inmutable de bytes. Representan una cadena de bytes en bruto y se utilizan principalmente para trabajar con datos binarios como archivos, sockets de red, criptografía, entre otros.

Los datos tipo bytes se pueden crear utilizando una sintaxis de prefijo b antes de la cadena de bytes o utilizando la función bytes(). Los elementos de los datos tipo bytes son enteros en el rango de 0 a 255.

* Literales

los literales son un tipo de datos que almacenan valores base para el código fuente del software, mientras que las variables son contenedores que almacenan valores de datos, y las constantes son variables que no pueden cambiar su valor:

los literales son notaciones para valores constantes de algunos tipos incorporados, como cadenas, números, enteros, flotantes, imaginarios, etc.

* Variables

Son contenedores que almacenan valores de datos, y su valor puede cambiar. En Python, no es necesario indicar explícitamente el tipo de variable que se quiere definir, ya que puede ser de cualquier tipo. Para crear una variable, se utiliza el operador de asignación (=) y se le asigna el valor deseado.

* Constantes

Son variables que no pueden cambiar su valor. En Python, las constantes se suelen declarar y asignar en un módulo, que es un nuevo archivo que contiene variables y funciones. Dentro del módulo, las constantes se escriben en letras mayúsculas y separadas las palabras con el carácter underscore (\_).

* Identificadores

Los identificadores son nombres que se le dan a variables, funciones, clases, módulos u otros objetos. Los identificadores pueden tener cualquier longitud, pero deben cumplir con las siguientes condiciones:

Comenzar con una letra (mayúscula o minúscula) o con un guion bajo (\_)

No contener signos de puntuación como @, $ y %, excepto el guion bajo (\_)

Todas las letras deben pertenecer al código de caracteres ASCII básico

Se recomienda nombrar las clases comenzando por una letra mayúscula y el resto de los identificadores por una letra minúscula.

Los identificadores que comienzan con un guion bajo o con dos guiones bajos son privados. Si el identificador finaliza con dos guiones bajos, es un nombre especial definido por lenguaje.

* Operadores

Python cuenta con diferentes tipos, entre ellos:

1. Operadores de comparación, que comparan dos expresiones
2. Operadores lógicos, que vinculan expresiones en un contexto booleano
3. Operadores bit a bit, que manipulan números enteros como secuencias binarias
4. Operadores de asignación, que asignan un nombre a un valor
5. Operador de identidad, que indica si dos variables hacen referencia al mismo objeto

* Salida de datos

La salida de datos se puede realizar con la función print(). Esta función puede tener un número variable de argumentos, y cuando estos son cadenas de caracteres, se muestran tal cual en la consola de salida.

Si el argumento de la función no es una cadena de caracteres, print() realiza la conversión requerida desde el tipo de dato original a str.

Para justificar el valor a la izquierda o a la derecha con un ancho de campo especificado, se pueden utilizar modificadores de formato.

Para realizar operaciones de entrada y salida de datos en Python, también se puede utilizar la función input(), que permite al usuario ingresar datos desde el teclado.