Aspectos básicos de Python

Como comentamos líneas

- Usamos # para una sola línea
- Usamos “”” al inicio y al final del texto “”” para múltiples líneas.

Operadores aritméticos

3 + 4 suma
3 - 4 resta
3 * 4 multiplicación
3 / 4 división
3 // 4 cociente --> divide dos números y redondea el resultado hacia abajo, al entero más cercano
3 % 4 modulo --> devuelve el residuo de una división
3 ** 4 potencia

Operadores relacionales

< → Devuelve True si el operador de la izquierda es mayor que el operador de la derecha 12 > 3 devuelve True.

< → Devuelve True si el operador de la derecha es mayor que el operador de la izquierda 12 < 3 devuelve False.

== → Devuelve True si ambos operandos son iguales 12 == 3 devuelve False.

>= → Devuelve True si el operador de la izquierda es mayor o igual que el operador de la derecha 12 >= 3 devuelve True.

<= → Devuelve True si el operador de la derecha es mayor o igual que el operador de la izquierda 12 <= 3 devuelve False.

!= → Devuelve True si ambos operandos no son iguales

Operadores Lógicos

Operador	Nombre	Ejemplo
and	Devuelve True si ambos elementos son True	True and True = True
or	Devuelve True si al menos un elemento es True	True or False = True
not	Devuelve el contrario, True si es Falso y viceversa	not True = False

Variables

Es contenedor o un espacio en la memoria física o virtual de una computadora, donde se almacenan distintos tipos de datos (valores) y sirven para guardar y recuperar datos, representar valores existentes y asignar unos nuevos.

Se caracteriza por

- El nombre único que la identifica dentro del programa
- El valor guardado que puede cambiar durante la ejecución del programa
- El tipo de datos que se almacena, como enteros (2), floats (1,5), strings (texto), booleanos (true, false).

Sintaxis

- Identificador (es único)
- Signo = que es palabra reservada para asignar un valor a una variable
- Expresión

Ejemplo: variable = 2,

Tipos de variables

- Simples:
  - num_of_cards = 56
- Múltiples:
  - str_red, str_blue = 'Red', 'Blue' (cada identificador obtiene una expresión en su orden de izquierda a derecha)
  - ltr1, ltr2, ltr3 = 'Red' (cada identificador obtiene la misma expresión)

Tipo de datos

- Enteros (int) --> 2
- Flotantes (float) --> '2.5
- Strings (textos o caracteres) --> 'hola' o se puede usar con doble comillas "hola", si a un int o float le ponemos comillas se vuelve sting
- Booleanos (bool) --> True o False

Buenas prácticas según el PEP 8 --> https://peps.python.org/pep-0008/

- Tienen que empezar por una letra o barra baja.
- El uso de keywords como nombres está prohibido.
- Los nombres deben de ser descriptivos.
- Deben de estar en minúsculas y separando palabras por barras bajas '_'.
- Las constantes se escriben en mayúsculas y SNAKE_CASE.

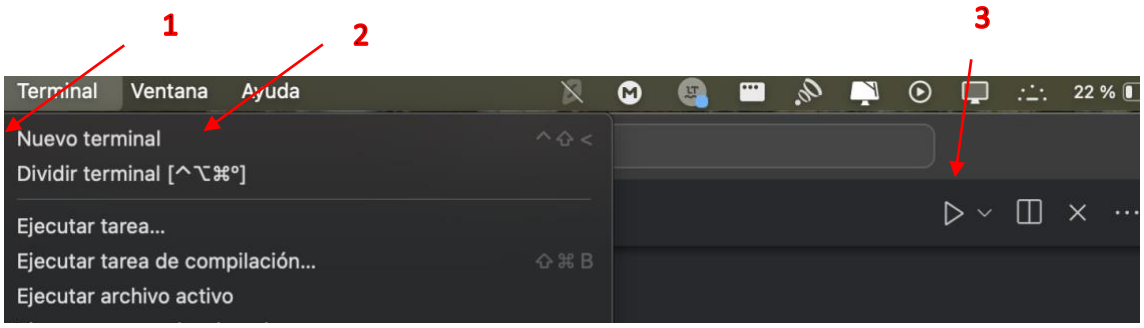
Escribamos nuestro primer hola mundo

Pero antes de escribir nuestro código tenemos que habilitar la terminal que nos permitirá ver los resultados de nuestras variables.

Para abrir la terminal seguiremos los siguientes pasos:

- Barra superior → buscamos terminal
- Seleccionamos nueva terminal → nos aparecerá abajo
- Para correr el código tenemos que buscar en la esquina superior derecha un icono de un triángulo, al presionarlo tendremos en consola el resultado.

Pasos para abrir terminal:



Para poder visualizar algo en pantalla escribimos la palabra reservada print(), dentro de las () tenemos que indicar que queremos mostrar.

Ejemplos:

- Sí, declaramos una variable

```
variable_hola = 'Hola Mundo'
print(variable_hola)
```

- Sin declarar variable

```
print('Hola Mundo')
```

Cómo consultar el tipo de dato

```
print(type("Soy un dato str"))  # Tipo 'str'
print(type(5))  # Tipo 'int'
print(type(1.5))  # Tipo 'float'
print(type(3 + 1j))  # Tipo 'complex'
print(type(True))  # Tipo 'bool'
print(type(print("Mi cadena de texto")))  # Tipo 'NoneType'
```

Concatenación de variables

print(my_string_variable, my_int_to_str_variable, my_bool_variable)
print("Este es el valor de:", my_bool_variable)

Algunas funciones del sistema

print(len(my_string_variable)). → permite saber el valor de los caracteres que tiene la variable

Inputs

Permite que se habilite una pestaña en pantalla para que podamos introducir un dato
name = input('¿Cuál es tu nombre? ')
age = input('¿Cuántos años tienes? ')

Listas

Es un tipo de dato que permite almacenar datos de cualquier tipo (int, float, string, booleans) son mutables y dinámicas.

Sintaxis:

lista = [1,2,3] o lista = list("1234)

Propiedades de las listas:

- Son ordenadas, mantienen el orden en el que han sido definidas
- Pueden ser formadas por tipos arbitrarios
- Pueden ser indexadas con [i].
- Se pueden anidar, es decir, meter una dentro de la otra.
- Son mutables, ya que sus elementos pueden ser modificados.
- Son dinámicas, porque se pueden añadir o eliminar elementos.

Tuplas

Permite almacenar datos de una manera muy parecida a las listas, con la salvedad de que son inmutables (no pueden ser modificadas una vez declaradas)
Diccionarios.

Sintaxis:

tupla = (1, 2, 3)

Diccionarios

Permite almacenar su contenido en forma de llave y valor. donde cada uno tiene una llave key y un valor value. Los diccionarios se pueden crear con paréntesis {} separando con una coma cada par key: value.

Sintaxis:

```
d1 = {
        "Nombre": "Sara",
         "Edad": 27,
        "Documento": 1003882
}
```

También es posible usar el constructor dict() para crear un diccionario.

```
d3 = dict (Nombre='Sara',
        Edad=27,
        Documento=1003882
)
```

Propiedades de los diccionarios en Python son las siguientes:

- Son dinámicos, pueden crecer o decrecer, se pueden añadir o eliminar elementos.
- Son indexados, los elementos del diccionario son accesibles a través del key.
- Y son anidados, un diccionario puede contener a otro diccionario en su campo value.

El próximo documento será un archivo py para que podamos poner en práctica lo que hemos visto aquí y aprendamos como se puede ingresar a estos datos.