Clase 2

IPC2

Conversión de tipos

Cada tipo en Python viene con un comando de biblioteca que convierte los valores de un tipo en el tipo asociado al comando.

- Convertir a cadena de texto (string): str()
- Convertir a entero: int()
- A punto flotante (números decimales): float()
- A booleano: bool()

```
>>> str(23)
'23'
>>> str(3.5)
'3.5'
>>> str(True)
'True'
>>> float(12)
12.0
>>> float('13.65')
13.65
>>> int('123')
123
>>> int(13.54)
13
>>> bool(0)
False
>>> bool('')
False
>>> bool('Hola')
True
>>> bool(-3.14)
True
```

Sentencias

Una sentencia es una unidad de código
que el intérprete de Python puede ejecutar.

```
Python 3.12.0 (tags/v3.12.0:0fb18b0, Oct  2 2023, 13:03:39) [MSC v.1935 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> ejemplo="hola"
>>> print(ejemplo)
hola
>>>
```

LAS SENTENCIAS DE ASIGNACIÓN NO PRODUCEN UN RESULTADO.

Sentencia break

La sentencia break nos permite alterar el comportamiento de los bucles while y for.

Concretamente, permite terminar con la ejecución del bucle.

Esto significa que una vez se encuentra la palabra break, el bucle se habrá terminado.

```
for i in range(5):
    print(i)
    break
```

Sentencia continue

Salta todo el código restante en la iteración actual y vuelve al principio en el caso de que aún queden iteraciones por completar.

```
cadena = 'Python'
for letra in cadena:
    if letra == 'P':
        continue
    print(letra)
```

Sentencia return

Permite terminar la ejecución de una función antes de alcanzar su final

```
def muestra_raiz_cuadrada(x):
    if x <= 0:
        print("Por favor, use sólo números positivos.")
        return

    resultado = x**0.5
    print("La raíz cuadrada de x es", resultado)
```

Entrada por teclado

Python que reciben entradas desde el teclado: raw_input e input.

```
a = input('Hola! Ingresa tu nombre: ')
```

Evaluación de expresiones

01

Una expresión es una combinación de valores, variables y operadores. Si digita una expresión en la línea de comandos, el intérprete la evalúa y despliega el resultado.

02

La evaluación de una expresión produce un valor

03

Un valor, por si mismo, se considera como una expresión, lo mismo ocurre para las variables.

Identación

Python utiliza la indentación para delimitar la estructura permitiendo establecer bloques de código. No existen comandos para finalizar las líneas ni llaves con las que delimitar el código.

Los únicos delimitadores existentes son los dos puntos (:) y la indentación del código.

```
if a>2:
    print('estoy en el if')
else:
    print('estoy en el else')
    print ('sigo estando en el else')
```

Ejecución condicional

Consiste básicamente en las siguientes partes principales:

- Una prueba que evalúa a verdadero o falso
- Un bloque de código que se ejecuta si la prueba es verdadero
- Un bloque opcional de código si la prueba es falsa

```
if condicional:
    codigo
else:
    codigo
```

Condicionales encadenadas

A veces hay más de dos posibilidades y necesitamos más de dos ramas. Una forma de expresarlo es una condicional encadenada:

```
x=1
y=1
if x < y:
    print (x, "es menor que", y)
elif x > y:
    print (x, "es mayor que", y)
else:
    print (x, "y", y, "son iguales")
```

Funciones

Una función es una secuencia de sentencias que realizan una operación y que reciben un nombre. Cuando se define una función, se especifica el nombre y la secuencia de sentencias. Más adelante, se puede “llamar” a la función por ese nombre.

```
>>> len('Hola, mundo')
11
>>>
```

```
>>> int(3.99999)
3
>>> int(-2.3)
-2
```

Funciones

```
import random

for i in range(10):
    x = random.random()
    print(x)
```

```
def sumados(a, b):
    suma = a + b
    return suma

x = sumados(3, 5)
print(x)
```

Strings

Una cadena es una secuencia de caracteres. Se puede acceder a los caracteres de uno en uno con el operador corchete

```
>>> fruta = 'banana'
>>> letra = fruta[1]
```

```
>>> print(letra)
a
```

Operadores entre cadenas de caracteres

ef

Concatenar

ESTE TÉRMINO SIGNIFICA JUNTAR CADENAS DE CARACTERES. SE REALIZA MEDIANTE EL OPERADOR DE SUMA (+).

```
mensaje1 = 'Hola' + ' ' + 'Mundo'
print(mensaje1)


mensaje3 = 'Hola'
mensaje3 += ' '
mensaje3 += 'Mundo'
print(mensaje3)
```

ef

Multiplicar

SI QUIERES VARIAS COPIAS DE UNA CADENA DE CARACTERES UTILIZA EL OPERADOR DE MULTIPLICACIÓN (*)

```
mensaje2a = 'Hola ' * 3
```

Métodos para cadenas de caracteres

ef

Extensión

DETERMINA EL NÚMERO DE CARACTERES EN UNA CADENA UTILIZANDO EL MÉTODO LEN

```
mensaje4 = 'hola' + ' ' + 'mundo'
print(len(mensaje4))
```

ef

Minúsculas

CONVIERTE UNA CADENA DE CARACTERES A MINÚSCULAS

```
mensaje7 = "HOLA MUNDO"
mensaje7a = mensaje7.lower()
print(mensaje7a)
```

ef

Reemplazar

PARA CAMBIAR UNA SUB-CADENA DE UNA CADENA SE PUEDE UTILIZAR EL MÉTODO REPLACE

```
mensaje8 = "HOLA MUNDO"
mensaje8a = mensaje7.replace("L", "pizza")
print(mensaje8a)
```

ef Cortar

```
mensaje9 = "Hola Mundo"
mensaje9a = mensaje9[1:8]
print(mensaje9a)
```

```
mensaje9 = "Hola Mundo"
startLoc = 2
endLoc = 8
mensaje9b = mensaje9[startLoc: endLoc]
print(mensaje9b)
```

ef

Encontrar

SE PUEDE BUSCAR UNA SUB-CADENA EN UNA CADENA DE CARACTERES UTILIZANDO EL MÉTODO FIND

```
mensaje5 = "Hola Mundo"
mensaje5a = mensaje5.find("Mundo")
print(mensaje5a)
```