

Python

ADA1: Investigación documental de lenguajes de programación.
Equipo: “Programadores Insomnes”

Integrantes:

+

Campos Daguer Emilio
Cough Martin Reynaldo
Graniel Arzat Aaron Isaac
Herrera Herrera Adiel Elio enai
Méndez Sierra Daniel

Instrucciones: En equipo, seleccionar un lenguaje de programación (Java, Python, C++, C#, JavaScript, VBScript, Ruby, etc.), realizar una investigación documental que sintetice la información que se solicita y elaborar una presentación (máximo 15 diapositivas).

Antecedentes de Python

Guido van Rossum el creador de este programa empezó a idearlo a finales de los 80 para implementarlo en 1989.

Posteriormente en 1991 sería liberada la versión para el público (versión 0.9.0).

En octubre del 2000 sería publicada la segunda versión.

La versión 3.0 sería publicada en diciembre de 2008.

Cuenta con estructuras de datos eficientes y de alto nivel, así como un **enfoque** simple pero efectivo hacia la programación orientada a objetos.

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional.

Principales palabras reservadas

False: Valor booleano para operaciones de comparación o lógicas que denota un valor falso.

None: Valor nulo

True: Valor booleano para operaciones comparativas o lógicas que denota un valor verdadero

And- Operador lógico

As- Alias

Assert- Utilizado con fines de depuración

Async- De la biblioteca “asyncio” de Python que es utilizado par código recurrente en el lenguaje

Await- De la biblioteca “asyncio” de Python que es utilizado par código recurrente en el lenguaje

Break- Alterar el comportamiento de bucles for y while

Class- Definir una nueva clase

Def- Definir una función

Del- Eliminar

Elif, Else, If – Declaraciones u operadores condicionales



Except, raise, try - Crear excepciones

Finally - Tienen la función de garantizar que un bloque de código se ejecute

For- Bucles

From- Importar partes específicas de un módulo

Global- Declarar variable global

Import- Importar un módulo

In- Comprobar si un valor está incluido

Is- Probar si dos variables se refieren al mismo objeto.

Lambda- Crear función anónima

Nonlocal- Declarar variable no local

Not- Operador lógico

Or- Operador lógico

Pass- Declaración nula utilizado como marcador de posición

Return- Para salir y devolver el valor a una función.

While- Crear bucles

With- Simplificar el manejo de excepciones

Yield- Para salir y devolver el valor a una función, con la excepción que devuelve un generador.



Tipos de variables más importantes.

int: representan números enteros, positivos y negativos.

```
year = 2021
dia = 7
edad = 26
temperatura = -5
angulo = -45
```

bool: datos binarios útiles para expresiones condicionales y comparaciones

```
esta_frio = True
es_bajo = False
```

float: representan números de coma flotante o decimales (números reales)

```
pi = 3.1416
estatura = 1.84
peso = 85.6
temperatura = -5.55
edad = 26.0
```

string: arreglo de caracteres que forman cadenas para formar un mensaje u oración generalmente. Se pueden crear usando comillas simples, dobles o triples.

```
profesor = "Sergio Castaño Giraldo"
web = 'Control Automático Educación'
cursos = """
1. Python
2. Matlab
3. MicroPython
4. Arduino
5. PIC
6. Control de Procesos
"""
```


Operadores

Aritméticos

+	Suma dos operandos
-	Resta al operando de la izquierda el valor del de la derecha. Utilizado sobre un único operando, le cambia el signo.
*	Producto de dos operandos
/	Divide el operando de la izquierda por el de la derecha. El resultado siempre es un float
%	Operador módulo. Obtiene el resto de dividir el operando de la izquierda por el de la derecha
//	Obtiene el cociente entero de dividir el operando de la izquierda por el de la derecha
**	Potencia. El resultado es el operando de la izquierda elevado a la potencia del operando de la derecha.

Relacionales

>	Mayor que. True si el operando de la izquierda es estrictamente mayor que el de la derecha; False en caso contrario.
>=	Mayor o igual que. True si el operando de la izquierda es mayor o igual que el de la derecha; False en caso contrario.
<	Menor que. True si el operando de la izquierda es estrictamente menor que el de la derecha; False en caso contrario.
<=	Menor o igual que. True si el operando de la izquierda es menor o igual que el de la derecha; False en caso contrario.
==	Igual. True si el operando de la izquierda es igual que el de la derecha; False en caso contrario.
!=	Distinto. True si los operandos son distintos; False en caso contrario.

Lógicos

IMPORTANTE: Las operaciones **and**, **or** y **not** realmente no devuelven True o False, sino que devuelven uno de los operandos como veremos en el cuadro de abajo.

Operación	Resultado	Descripción
<code>a or b</code>	Si <code>a</code> se evalúa a falso, entonces devuelve <code>b</code> , si no devuelve <code>a</code>	Solo se evalúa el segundo operando si el primero es falso
<code>a and b</code>	Si <code>a</code> se evalúa a falso, entonces devuelve <code>a</code> , si no devuelve <code>b</code>	Solo se evalúa el segundo operando si el primero es verdadero
<code>not a</code>	Si <code>a</code> se evalúa a falso, entonces devuelve <code>True</code> , si no devuelve <code>False</code>	Tiene menos prioridad que otros operadores no booleanos

Instrucciones de control de flujo

Condicionales:

Permiten comprobar condiciones para ejecutar o no acciones.
Analizan si es verdadera o falsa la condición (True/False).

(If, Elif, Else)

```
1 compra = 160
2 if compra >= 300:
3     print ("20% de descuento")
4     compra = compra*0.8
5 elif compra > 150 and compra < 300:
6     print ("10% de descuento")
7     compra = compra*0.9
8 else:
9     print ("No aplica descuento")
```

10% de descuento

>

Instrucciones de control de flujo

Repetitivas:

Permiten ejecutar un mismo código, o fragmento del mismo, repetidas veces mientras se cumpla una condición o iterar sobre una variable compleja

(While, For)

```
1 x = 1
2 while x <= 5:
3     print ("Prueba #", x)
4     x += 1
5
6 lista_1 = ['aguacate', 'tomate', 'perejil']
7 for z in lista_1:
8     print (z)
9
10 for z in "abc":
11     print (z)
```

```
Prueba # 1
Prueba # 2
Prueba # 3
Prueba # 4
Prueba # 5
aguacate
tomate
perejil
a
b
c
```

Bibliotecas/Funciones más relevantes.



Librería que te permite hacer peticiones por http de una manera sencilla.



Herramienta que agrega soporte para abrir, manipular y guardar muchos formatos de imágenes diferentes.



Scrapy es un framework que te permitirá rastrear sitios web y extraer datos estructurados.



Biblioteca de código abierto que proporciona estructuras de datos y herramientas de análisis de datos.

Bibliotecas/Funciones más relevantes.

The logo for matplotlib, featuring the word "matplotlib" in a blue sans-serif font. The "plot" part is replaced by a circular icon containing a stylized line graph with multiple colored lines (red, green, blue, yellow) and a black dot.

Biblioteca que produce gráficas de buena calidad en una variedad de formatos y entornos interactivos.

The logo for NumPy, featuring a blue 3D cube icon to the left of the word "NumPy" in a blue sans-serif font.

Biblioteca más usada para computación científica, contiene funciones sofisticadas, como herramientas para integración de código, etc.



Plataforma líder para la creación de programas en Python para trabajar con datos de lenguaje humano.

Funciones básicas de entrada y salida.

Entrada de datos con input:

Permite obtener información de la terminal escrita el teclado, se ingresa el texto que se necesite y se presiona la tecla enter.

```
nombre = "Pepe"  
edad = 25  
print("Me llamo", nombre, "y tengo", edad, "años.")
```

```
print("¿Cómo se llama?")  
nombre = input()  
print(f"Me alegro de conocerle, {nombre}")
```

Salida de datos con print:

Permite mostrar información por consola en la pantalla como mensajes, números o valores de una variable. El texto a mostrar se escribe como argumento de la función.

Ejercicio 1

```
1  # Ejercicio_Promedio
2
3  num1 = float(input("Escriba un número: "))
4  num2 = float(input("Escriba un número: "))
5
6  promedio = (num1+num2)/2
7
8  print(promedio)
```

Ejercicio 2

```
1  #Ejercicio_While
2
3  num = 0
4
5  While num<10
6      |   contador = num + 1
7      |
8  print(contador)
```


Referencias

- + <https://www.mclibre.org/consultar/python/otros/historia.html>
- + <https://www.iartificial.net/librerias-de-python-para-machine-learning/>
- + <https://www.mclibre.org/consultar/python/lecciones/python-entrada-teclado.html>
- + <https://www.mclibre.org/consultar/python/lecciones/python-salida-pantalla.html>
- + <https://j2logo.com/python/tutorial/operadores-en-python/#operadores-logicos>
- + <https://www.codigofuente.org/variables-en-python/>
<https://eiposgrados.com/blog-python/palabras-reservadas-python/>