



Python 101

By: Jose Simmonds

Quien Soy? 🤖

- Estudiante de Ing. Electromecánica
- Estudiante de Ing. Mecatrónica
- Junior Data Analyst
- Maker
- Tutor



COBOTS LAB: Curso de Introducción a lo Programación en Python. | 2023



Porque Aprender Python

- Sintaxis simple e intuitiva
- Lenguaje en expansion con gran futuro
- Libre y multiplataforma
- Aplicado a la industria y a la investigación

Fases del Curso

- Fundamentos de programación
- Manejo de datos
- Programación Orientada a objetos
- Temas avanzados

Fundamentos

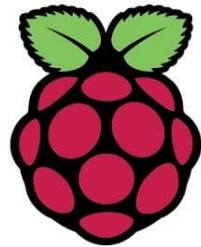
- Tipos de datos
- Expresiones
- Control de flujo

Manejo de datos

- Colecciones de datos
- Entradas y salidas
- Programacion de funciones
- Manejo de excepciones

Orientación a Objetos

- En Python (y otros lenguajes de programación de Raspberry Pi), casi todo el código que encontrarás se crea en un estilo llamado "programación orientada a objetos", o OOP para abreviar.
- El código se utiliza para crear objetos. Estos representan cosas del mundo real: un perro, una silla o las ruedas de un automóvil.

The background of the slide features a Raspberry Pi board and a computer screen displaying a code editor with Python code. A red banner is overlaid at the bottom.

**Object-oriented
programming in Python**

Temas avanzados

- Pandas
- Numpy
- OpenCV
- Matplotlib

Agenda

- Que es Python?
- Porque aprender Python ?
- Que es Google Colab? y Colab Notebook?
- Instalación de Python
- Entorno de Programación
- Variables, Tipos de Datos y Operadores en Python
- Estructura de Datos y Tipos de Datos (Nivel I)
- Write Your First Python Program!

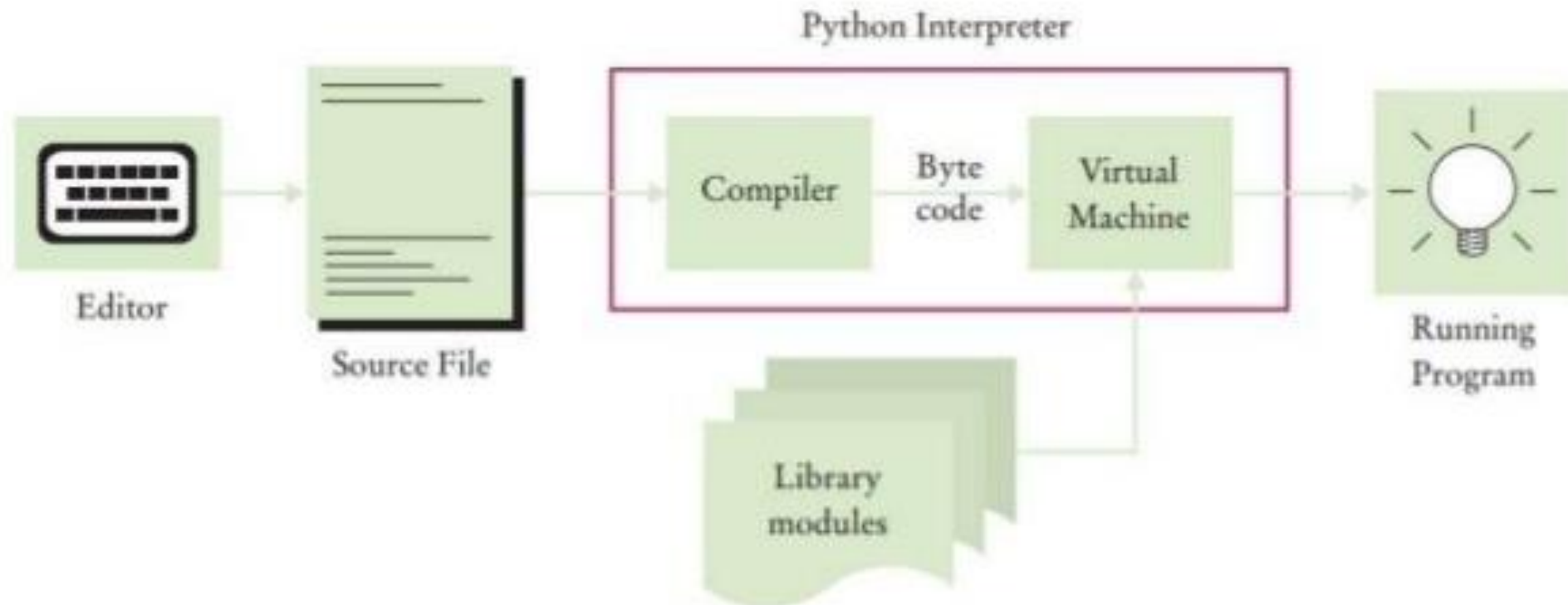
Origen de Python ☎ ☁

- Python fue concebido en los 1980s por Guido Van Rossum en el Centrum Wiskunde&Informatica (CWI) en Holanda y su implementacion comenzo en 1989



Que es Python? 🤔

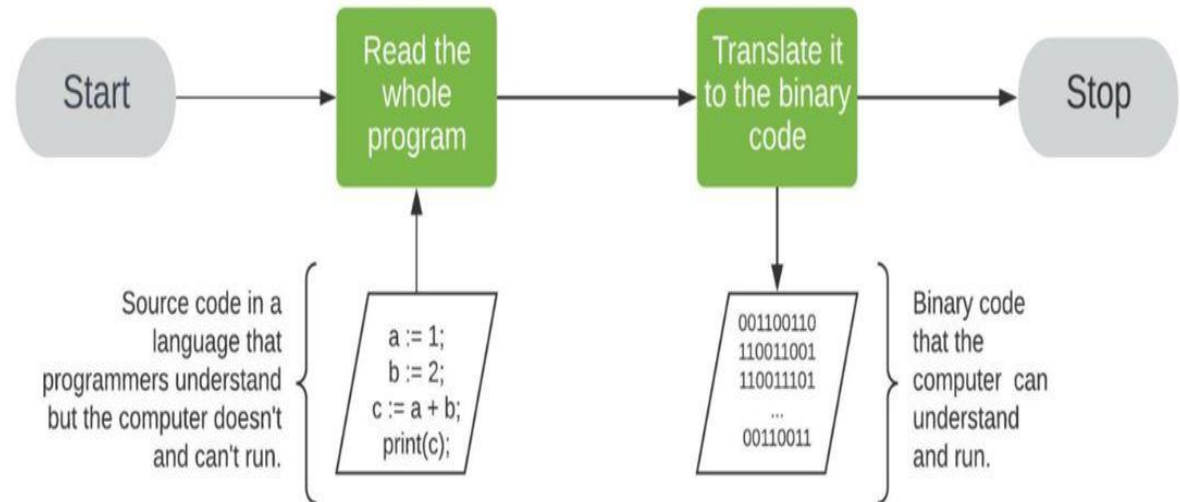
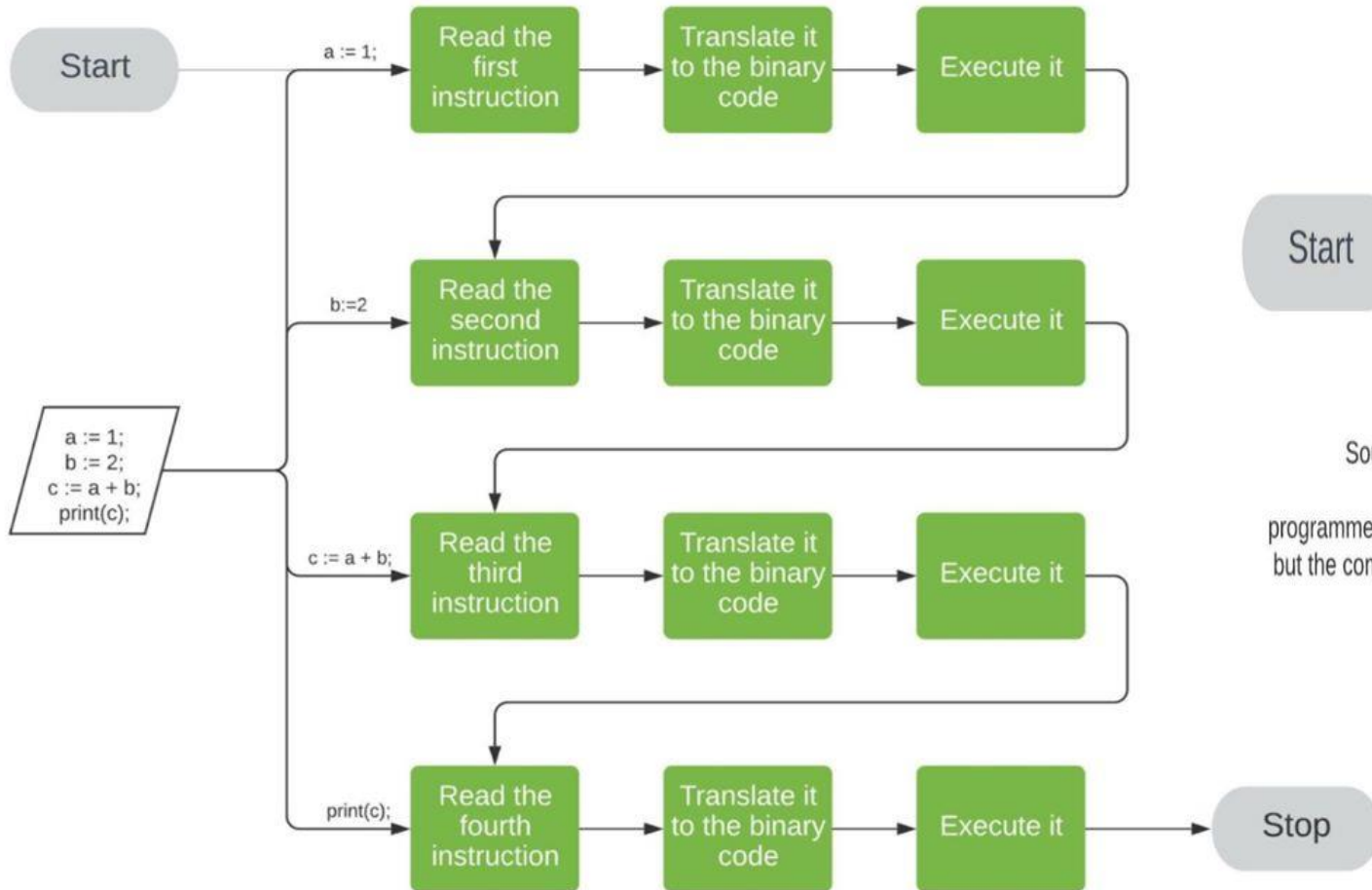
- Python es un lenguaje interpretado , que significa esto?
“Que no requiere compilación y se ejecuta según se va interpretando”



Ejemplo

```
Python 3.9.6 (default, Jun 29 2021, 16:24:54)
[GCC 10.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> # Hello, World! in Python
>>> print("Hello, World!")
Hello, World!
>>> exit()
```

Lenguaje Interpretado vs Lenguaje Compilado



Porque Aprender Python? 😊

- Libre y de Codigo Abierto
- Gran Comunidad
- Muchas bibliotecas para Ciencia de Datos “Data Science” y Aprendizaje Automatico “Machine Learning”.
- Facil e intuitivo de usar

Algunas librerías Importantes 🙌

- Pandas: Pandas es una de las bibliotecas de análisis y manipulación de datos más populares . Es una biblioteca de código abierto .
- Numpy: NumPy es una biblioteca de código abierto. Es una increíble biblioteca de Python para cálculos científicos. También permite realizar operaciones matriciales .
- Matplotlib: Matplotlib es una de las bibliotecas gráficas de Python en 2D más famosas utilizadas para la visualización de datos .

Otras librerías

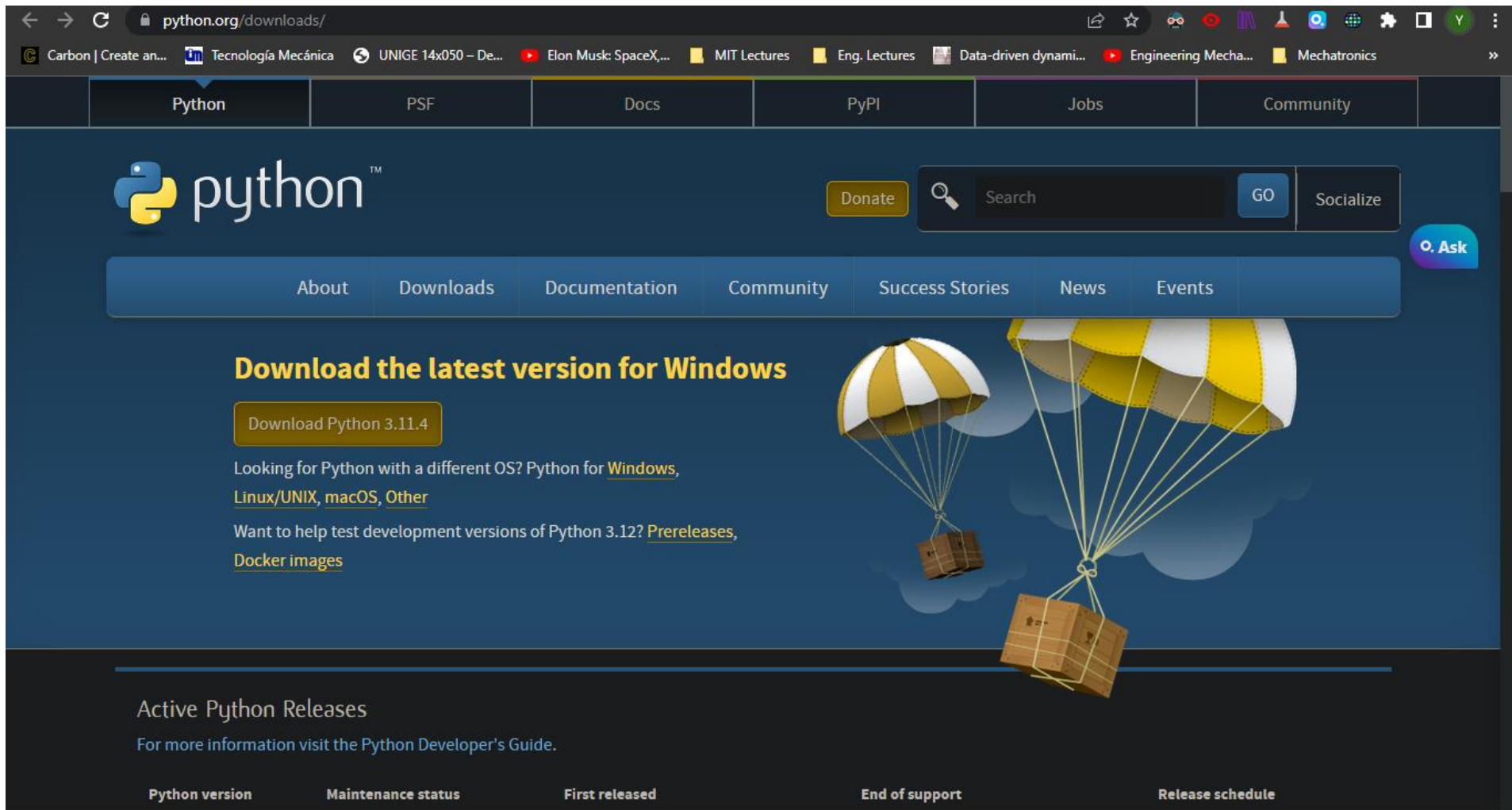


Que es Google Colab y Colab Notebook

- Entorno en línea para ejecutar código Python
- Google colab es la forma más sencilla de configurar un entorno de cuaderno interactivo para realizar sus tareas de ciencia de datos.
- El entorno le ofrece hasta 12 GB de RAM/GPU, etc. de forma gratuita y 100 GB de almacenamiento.
- No necesita preocuparse por instalar una aplicación Python voluminosa en su computadora portátil/computadora
- Enlace para acceder a Google colab:
<https://colab.research.google.com/>

Instalación de Python

<https://www.python.org/downloads/>



The screenshot shows the Python.org website's download page. The browser's address bar displays 'python.org/downloads/'. The page features a dark blue header with the Python logo and navigation links: Python, PSF, Docs, PyPI, Jobs, and Community. A search bar with a 'GO' button and a 'Socialize' link are also present. Below the header, a secondary navigation bar includes links for About, Downloads, Documentation, Community, Success Stories, News, and Events. The main content area is titled 'Download the latest version for Windows' and features a prominent yellow button labeled 'Download Python 3.11.4'. To the right of this button, there is an illustration of two parachutes carrying boxes. Below the button, text provides links for other operating systems: 'Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [macOS](#), [Other](#)'. Further down, it mentions 'Want to help test development versions of Python 3.12? [Prereleases](#), [Docker images](#)'. At the bottom, a section titled 'Active Python Releases' directs users to the 'Python Developer's Guide'. The footer contains a table with headers: 'Python version', 'Maintenance status', 'First released', 'End of support', and 'Release schedule'.

python.org/downloads/

Carbon | Create an... Tecnología Mecánica UNIGE 14x050 – De... Elon Musk: SpaceX... MIT Lectures Eng. Lectures Data-driven dynami... Engineering Mecha... Mechatronics

Python PSF Docs PyPI Jobs Community

python™

Donate Search GO Socialize

Ask

About Downloads Documentation Community Success Stories News Events

Download the latest version for Windows

Download Python 3.11.4

Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [macOS](#), [Other](#)

Want to help test development versions of Python 3.12? [Prereleases](#), [Docker images](#)

Active Python Releases

For more information visit the [Python Developer's Guide](#).

Python version	Maintenance status	First released	End of support	Release schedule
----------------	--------------------	----------------	----------------	------------------

Entorno de Programación

- IDE “Integrated Development Environment”

PyCharm



Visual
Studio Code



Sublime Text



Vim



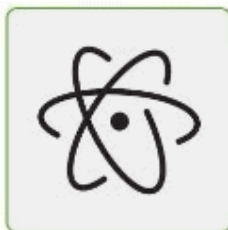
GNU Emacs



Spyder



Atom



Jupyter



Eclipse



IntelliJ IDEA



Notepad++



Variables, Tipos de Datos y Operadores en Python

- Comentarios en Python

- Python ignora todo después de la marca hash y hasta el final de la línea.
¡Puedes insertarlos en cualquier parte de tu código!
- Un acceso directo para agregar comentarios es mediante CTRL + /

```
In [14]: #assign a value 5 to b  
        b = 5
```

```
In [12]: b
```

```
Out[12]: 5
```

To write a comment
in Python, simply
put # before your
desired comment

Operadores en Python

- Los operadores se utilizan para realizar operaciones simples como sumas, comparaciones, etc. en variables y valores.
- Python admite los siguientes tipos de operadores, veremos algunos de los operadores más utilizados.

Operador	Descripción	Ejemplo
+	Suma	<pre>>>> 3 + 2 5</pre>
-	Resta	<pre>>>> 4 - 7 -3</pre>
-	Negación	<pre>>>> -7 -7</pre>
*	Multipliación	<pre>>>> 2 * 6 12</pre>
**	Exponente	<pre>>>> 2 ** 6 64</pre>
/	División	<pre>>>> 3.5 / 2 1.75</pre>
//	División entera	<pre>>>> 3.5 // 2 1.0</pre>
%	Módulo	<pre>>>> 7 % 2 1</pre>

Operadores aritméticos

- Podemos realizar algunas operaciones aritméticas básicas como sumas, restas, multiplicaciones, etc. utilizando Python.
- Ejemplo:
 - $x = 2$
 - $y = 3$
 - $x + y$ # suma
 - $x * y$ # multiplicación

```
In [2]: 4**2 #Exponentiation
```

```
Out[2]: 16
```

```
In [3]: 18%7 #Modulo
```

```
Out[3]: 4
```

Operadores Aritméticos: Significados y Ejemplos

Operator	Meaning	Example
+	Add two operands or unary plus	$x + y + 2$
-	Subtract right operand from the left or unary minus	$x - y - 2$
*	Multiply two operands	$x * y$
/	Divide left operand by the right one (always results into float)	x / y
%	Modulus - remainder of the division of left operand by the right	$x \% y$ (remainder of x/y)
//	Floor division - division that results into whole number adjusted to the left in the number line	$x // y$
**	Exponent - left operand raised to the power of right	$x ** y$ (x to the power y)

Operador de comparación

- Se utilizan para comparar dos valores.
- Da un resultado booleano (Verdadero/Falso)

Numeric Calculations:

```
In [1]: 2 < 3
```

```
Out[1]: True
```

```
In [2]: 2==3
```

```
Out[2]: False
```

```
In [3]: 2 <= 3
```

```
Out[3]: True
```

Other Comparisons:

```
In [4]: "rahuł" < "rohan"
```

```
Out[4]: True
```


Operadores de comparación: significados y ejemplos

Operator	Meaning	Example
>	Greater than - True if left operand is greater than the right	<code>x > y</code>
<	Less than - True if left operand is less than the right	<code>x < y</code>
==	Equal to - True if both operands are equal	<code>x == y</code>
!=	Not equal to - True if operands are not equal	<code>x != y</code>
>=	Greater than or equal to - True if left operand is greater than or equal to the right	<code>x >= y</code>
<=	Less than or equal to - True if left operand is less than or equal to the right	<code>x <= y</code>

Operador lógico

- Los operadores lógicos se utilizan para combinar sentencias condicionales.
- Da un resultado booleano (Verdadero/Falso)

Operator	Description
and	Returns True if both statements are true
or	Returns True if one of the statements is true
not	Reverse the result, returns False if the result is true

```
In [1]: x = 6  
x < 5 and x < 10
```

```
Out[1]: False
```

```
In [3]: x < 10 or x < 4
```

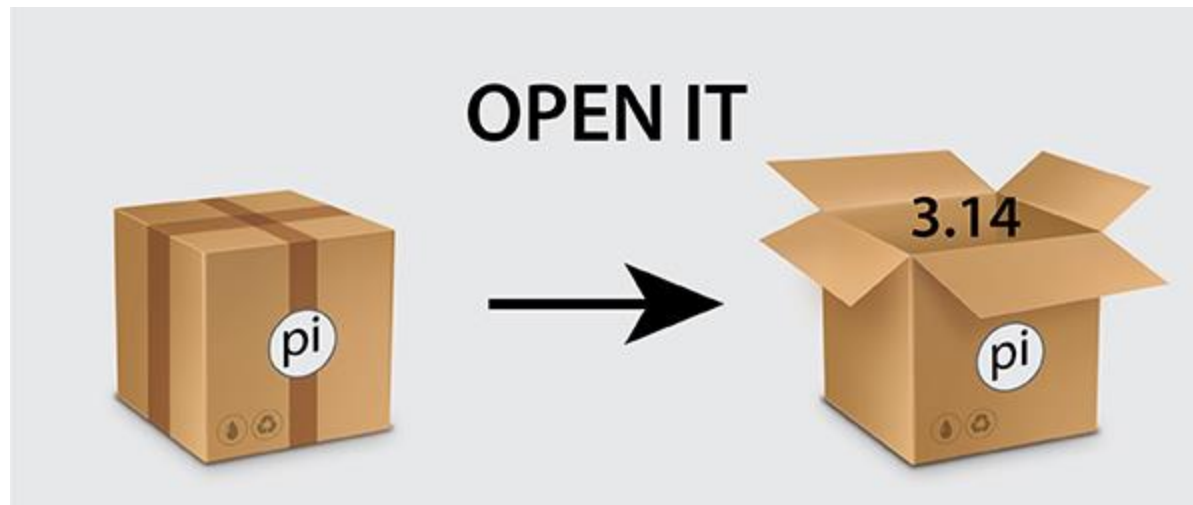
```
Out[3]: True
```

```
In [4]: not(x < 5 and x < 10)
```

```
Out[4]: True
```

Variables

- Una variable puede considerarse como un contenedor de almacenamiento de datos.
- Cada variable tendrá un nombre.
- Es una buena manera de almacenar información y, al mismo tiempo, facilitar la referencia posterior a esa información en nuestro código.

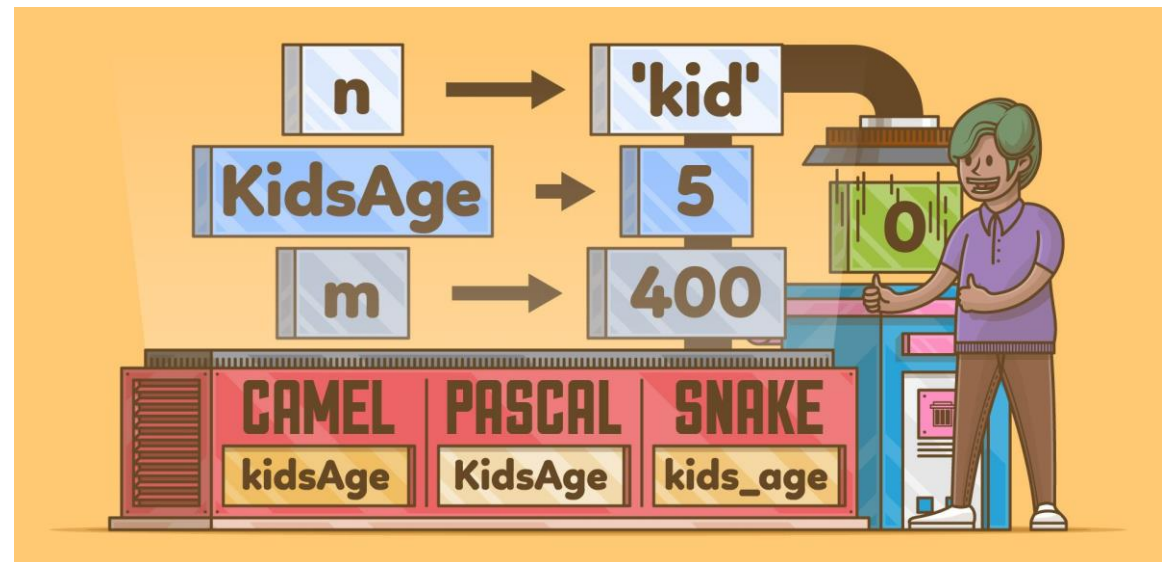


- Por ejemplo, en lugar de trabajar con el número 3.14, podemos asignarlo a una variable pi y usarlo tantas veces como queramos en nuestro código (cuando sea necesario).
- El signo igual (=) se utiliza para asignar valores a las variables.
- La sintaxis para asignar valores a una variable es la siguiente: Nombre de variable = valor o información.

• Ejemplo:

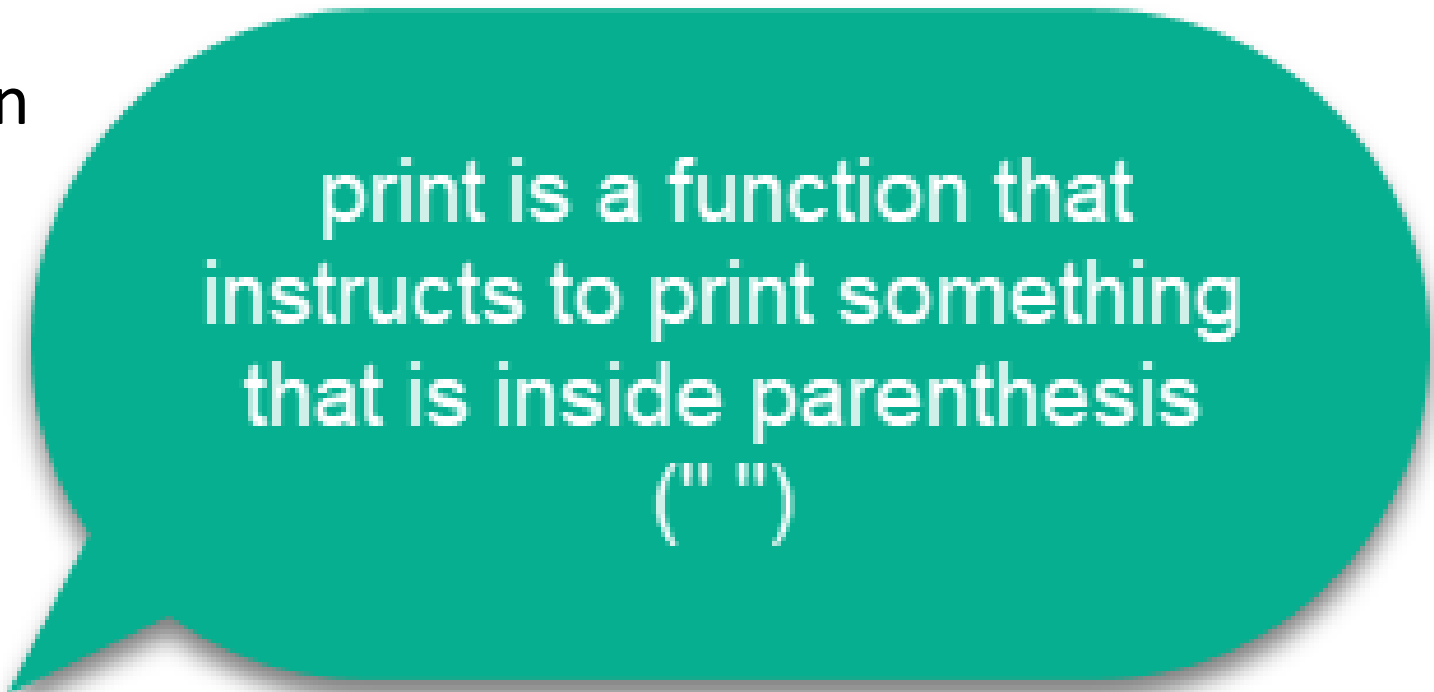
`x = 5`

`y = "Juan"`



Entrada y salida en Python

- Función de impresión



print is a function that
instructs to print something
that is inside parenthesis
(" ")

```
In [5]: print("Hello world")
```

```
Hello world
```

- Función de entrada

similar to print, input is also a function. Input function instructs to get an answer for a question asked in parenthesis (" ")

```
In [7]: x = input("What is your age?")
```

```
What is your age?18
```

```
In [8]: type(x)
```

```
Out[8]: str
```

By default the data type of the input taken from the user is 'str' i.e. string. One can do type conversion to integer as shown below

Quiz 🤖

- <https://www.menti.com/almxdie7bzk3>
- The voting code **6508 2263**



Exercise - Coding Challenge

- Considere dos números, $a = 5$, $b = 16$. Escribe un programa en Python para imprimir la suma de estos dos números.
- Escriba un programa Python para tomar un número del usuario como entrada e imprimir el cubo del número.

Strings

- Tipos de secuencia
- Las secuencias permiten almacenar múltiples valores de forma organizada y eficiente.
- Existen siete tipos de secuencia: strings, Unicode strings, lists, tuples, bytearrays, buffers, and xrange objects.
- No hay de qué preocuparse viendo esta larga lista, la irás conociendo poco a poco.

Python Strings

- Las cadenas son secuencias de caracteres.
- Veamos algunos ejemplos de cadenas: "Mi nombre es Rahul", "Rahul", "Vete a casa". Todos estos son ejemplos de cadenas.
- En Python, las cadenas se llaman str.
- Hay una forma específica de definir String en Python - se define entre comillas simples (') o dobles (") o incluso triples (""").

Accessing String Elements

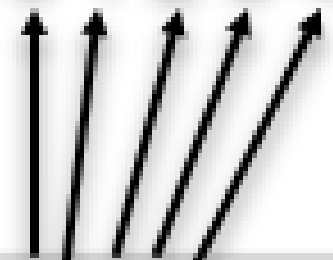
- Los corchetes pueden utilizarse para acceder a elementos de la cadena.
- Recuerde que el primer carácter tiene índice 0.
- El índice se refiere a la posición de un carácter en una cadena. En python el índice empieza por 0.

Ejemplo

```
1 a = "Hello, World!"
```

```
1 print(a[1])
```

index = 0 1 2 3 4



```
a = "Hello, World!"  
print(a[1])
```

String Slicing

- También podemos llamar a un rango de caracteres de la cadena utilizando el troceado de cadenas.
- Especifique el índice inicial y el índice final, separados por dos puntos, para devolver una parte de la cadena. Tenga en cuenta que el carácter del índice final no se incluye.
- Supongamos que queremos imprimir Mundo a partir de la cadena "Hola Mundo". Podemos hacerlo como se indica a continuación:

```
In [3]: a = "Hello, World!"  
        print(a[7:12])
```

```
World
```

Negative Indexing

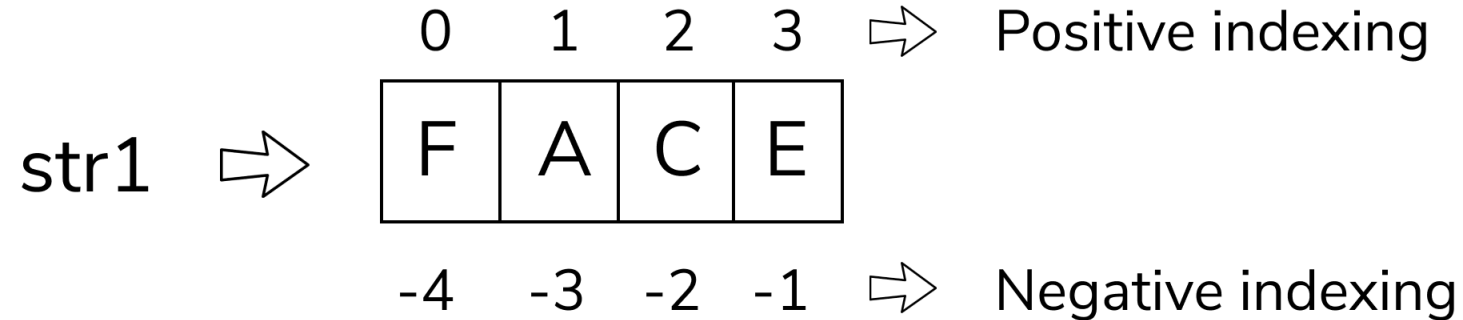
- Si tenemos una cadena larga y queremos localizar un elemento hacia el final, también podemos contar hacia atrás desde el final de la cadena, empezando en el índice -1.

```
1 a = "Hello, World!"  
  
1 print(a[-4])
```

- Obtiene los caracteres desde la posición -5 hasta la posición -3, comenzando la cuenta desde el final de la cadena:

```
1 print(a[-5:-2])
```

Output: orl



str1[1:3] = AC

str1[-3:-1] = AC


String Concatenation

- La concatenación de cadenas significa sumar cadenas.
- Utilice el carácter + para añadir una variable a otra variable:

```
In [9]: 'ab' + 'cd' #concatenation
```

```
Out[9]: 'abcd'
```


Otro Ejemplo



```
1  # Concatenating a String and an Int in Python
2  word = 'datagy'
3  integer = 2022
4
5  # Use + and str()
6  new_word = word + str(integer)
7  print(new_word)
8  # Returns: datagy2022
9
10 # Use f-strings
11 new_word2 = f'{word}{integer}'
12 # Returns: datagy2022
13
```

String Concatenation: Add Space

```
In [13]: x = 'ab'  
        y = 'cd'
```

```
In [14]: # Adding spaces between two strings  
        print(x + " " + y)
```

```
ab cd
```

String Length

- Para obtener la longitud de una cadena, utilice la función `len()`.

Getting length of the string a :

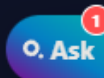
```
1 a = "Hello, World!"
```

```
1 print(len(a))
```

Output: 13

Strings Methods

- Python tiene un conjunto de métodos incorporados que puede utilizar en cadenas.
- Debes aprender: Aprende sobre los métodos de cadena más importantes en la siguiente hoja de trucos: [Python String Methods Cheatsheet](#)
- Consejo: Si no puedes seguirlo, ejecuta el código y descubre la diferencia.



Strings

↓ Print Cheatsheet

🔗 Share ▾

TOPICS

Hello World

Control Flow

Lists

Loops

Functions

Python: Code Challenges
(Optional)

Strings

Modules

Strings

In computer science, sequences of characters are referred to as *strings*. Strings can be any length and can include any character such as letters, numbers, symbols, and whitespace (spaces, tabs, new lines).

Escaping Characters

Backslashes (\) are used to escape characters if

```
txt = "She said \"Never let go\"."
```

Ejemplos de Strings Methods

Upper:

```
>>> w='hello!'
>>> w.upper()
'HELLO!'
```

Lower:

```
>>> w='HELLO!'
>>> w.lower()
'hello!'
```

Replace:

```
>>> r='rule'
>>> r.replace('r','m')
'mule'
```

Count:

```
>>> numbers=['1','2','1','2','2']
>>> numbers.count('2')
3
```

Quiz 🤖

- <https://www.menti.com/alu5byw>
- The voting code **6798 1834**



Tarea

Ejercicio #1

Tome el nombre y apellido como entrada y almacénelo en las variables `first_name` y `last_name`.

Supongamos

```
que first_name = "abc" last_name = "xyz",
```

Su tarea es imprimir lo siguiente:

Hola abc xyz! Acabas de profundizar en python

Ejercicio #2

Crear un programa de saludo

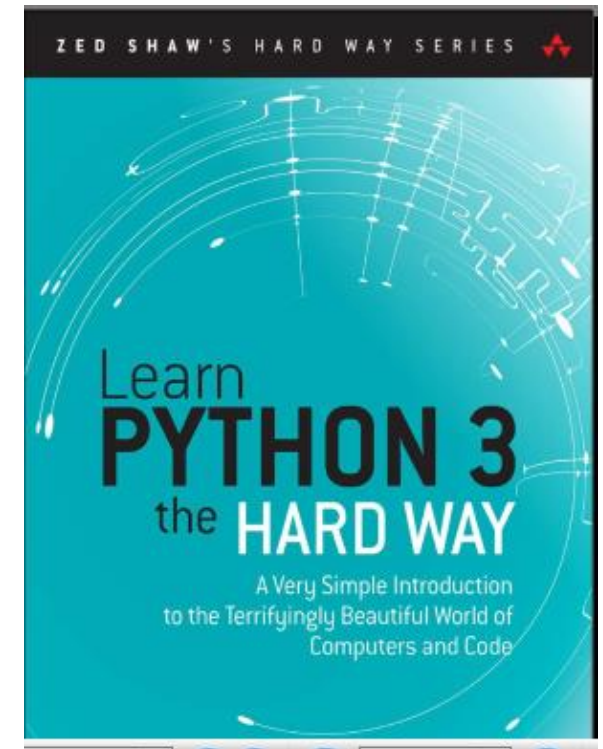
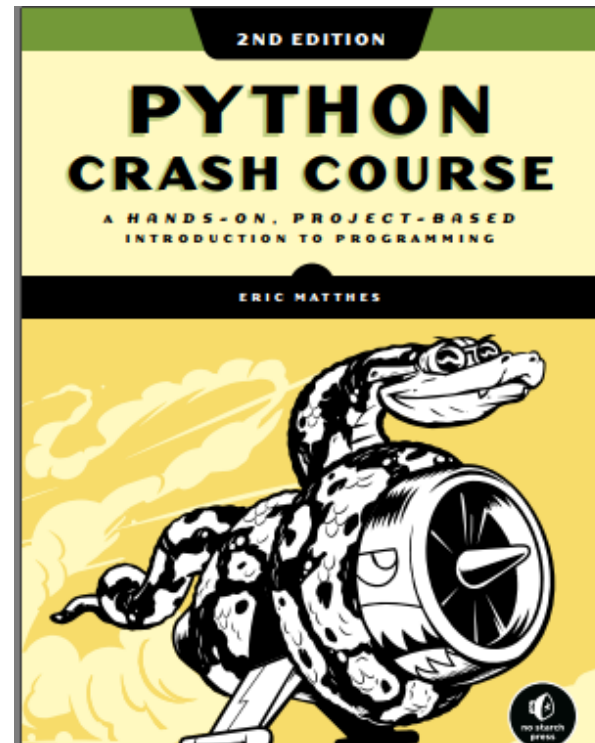
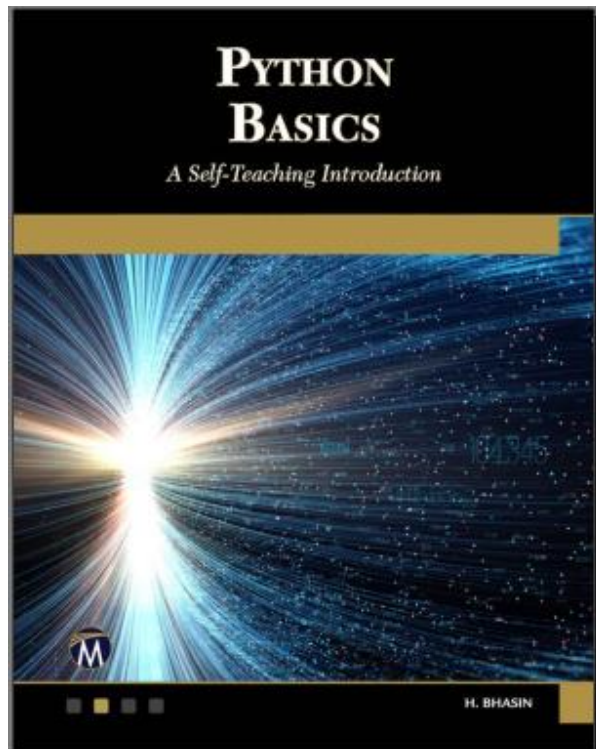
1. Preguntar al usuario por la ciudad en la que creció
2. Preguntar al usuario el nombre de una mascota
3. Combina el nombre de su ciudad y de su mascota y muéstrale el nombre de su banda

Ejercicio #3

3. Construir una calculadora de propinas

Libros y webs para consulta

- <https://www.w3schools.com/python/>
- <https://realpython.com/>
- <https://www.freecodecamp.org/news/tag/python/>
- <https://www.python.org/about/gettingstarted/>



```
While(noSuccess)
```

```
{
```

```
tryAgain();
```

```
if(success)
```

```
{
```

```
improve()
```

```
}
```

```
};
```

Nos Vemos en la proxima clase! 😊