Hablemos de

Python

¿Qué necesitamos para iniciar con Python?

ó

https://www.online-python.com/

Instalación

Windows:

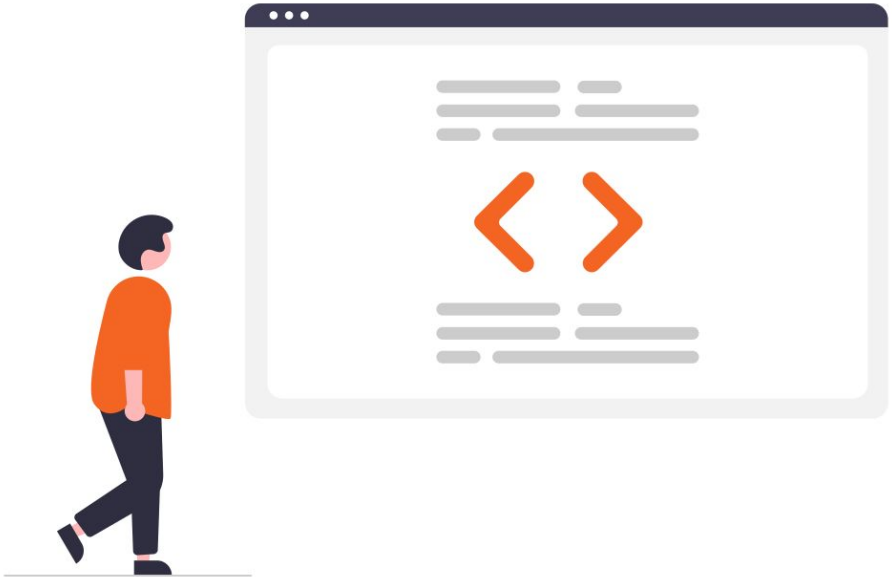
Descarga la última versión de Python para Windows desde la página web oficial:
https://www.python.org/downloads/windows/

Ubuntu:

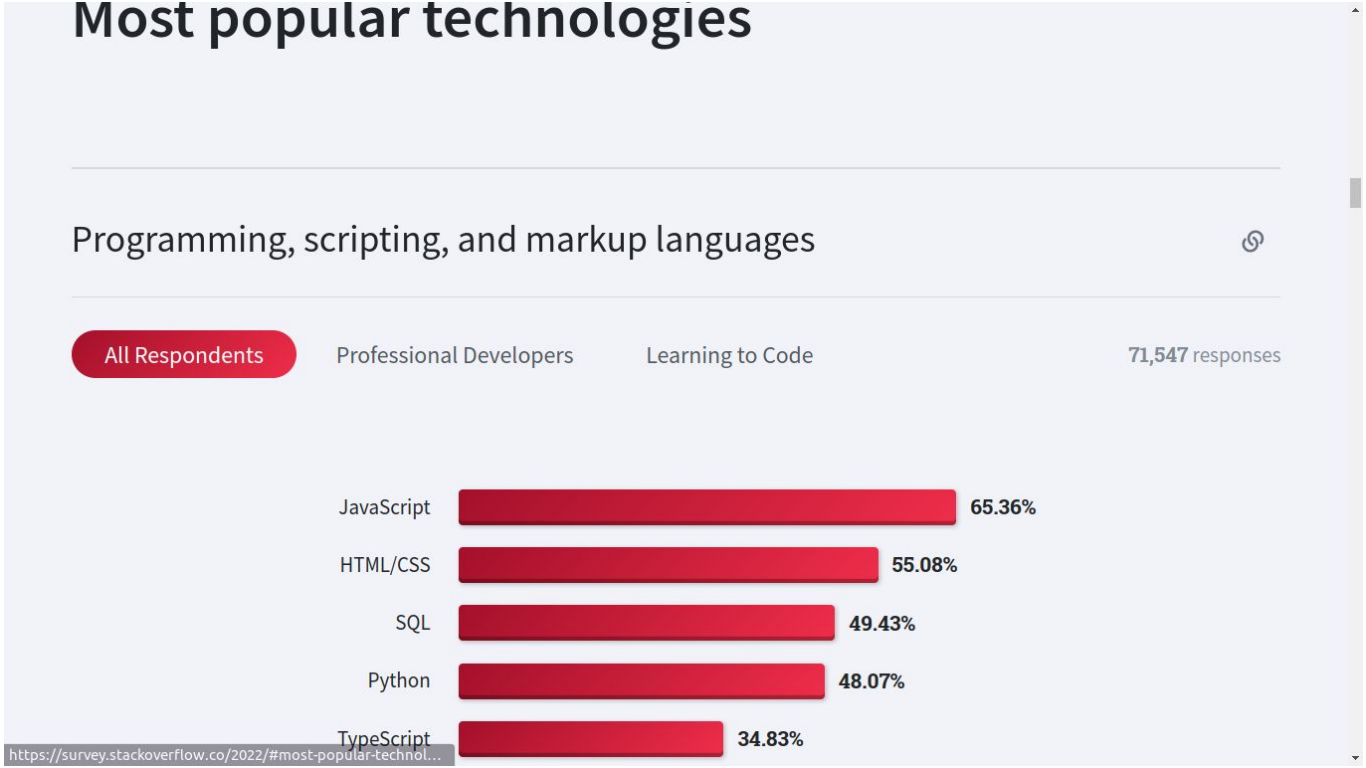
Abrir una terminal y ejecutar el siguiente comando:

sudo apt install python3

Introducción a Python

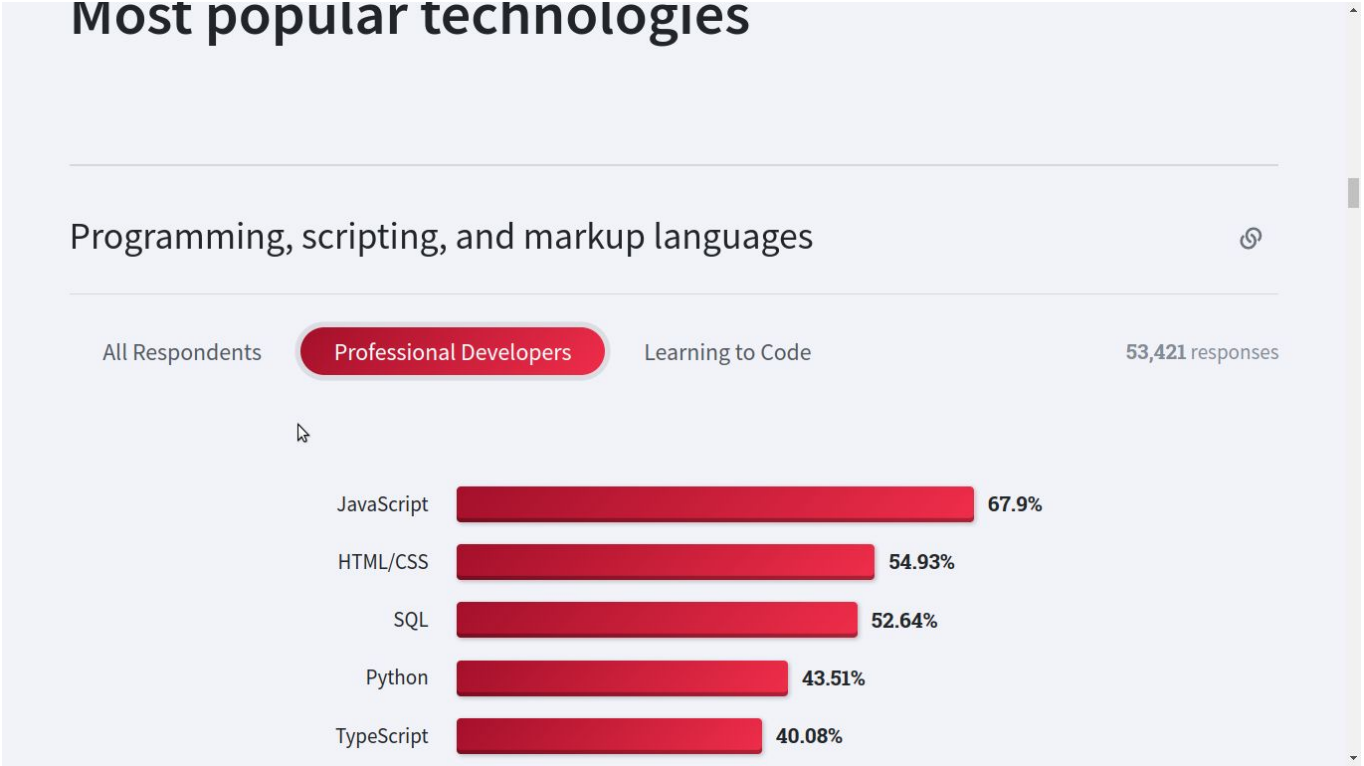


- Un lenguaje muy popular.



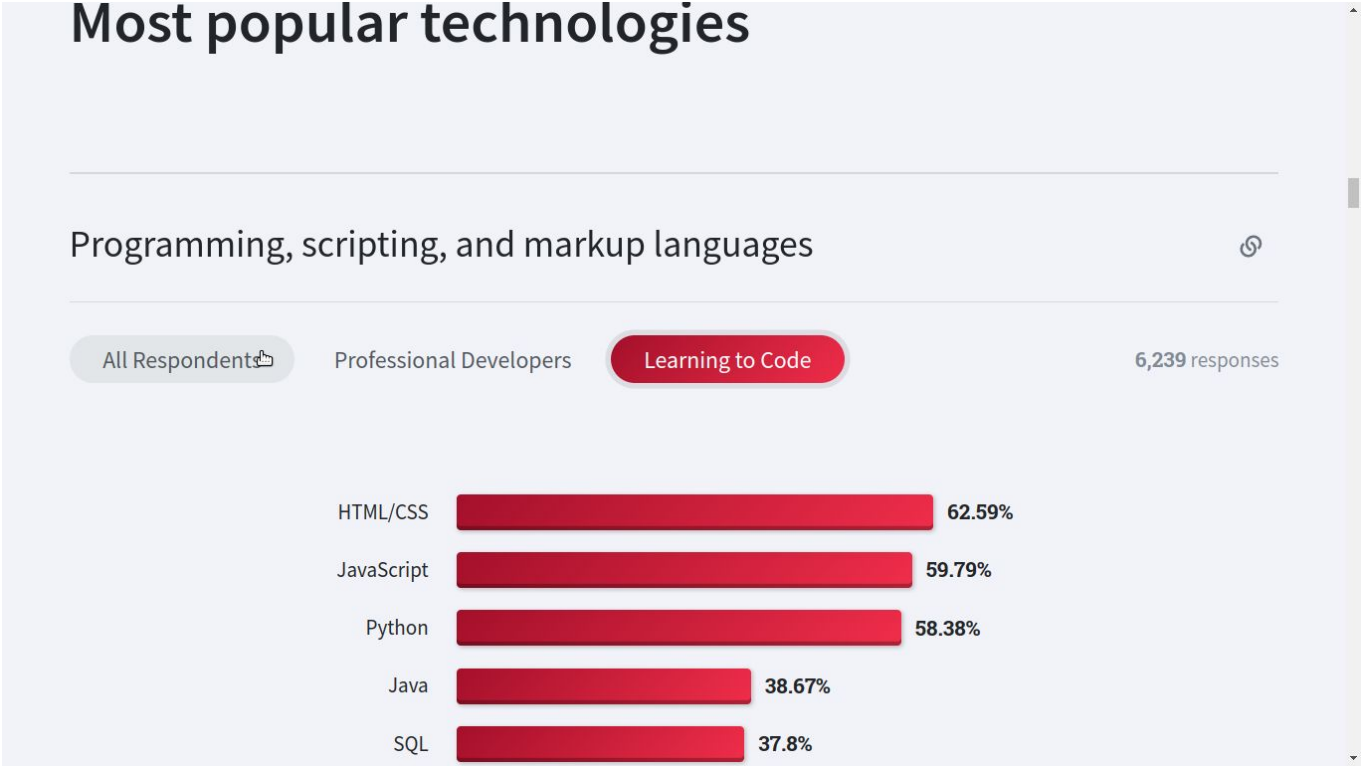
Extraído de la “Encuesta para desarrolladores de Stack Overflow 2022”
https://survey.stackoverflow.co/2022/#section-most-popular-technologies-programming-scripting-and-markup-languages

- Un lenguaje muy popular.



Extraído de la “Encuesta para desarrolladores de Stack Overflow 2022”
https://survey.stackoverflow.co/2022/#section-most-popular-technologies-programming-scripting-and-markup-languages

- Un lenguaje muy popular.



Extraído de la “Encuesta para desarrolladores de Stack Overflow 2022”
https://survey.stackoverflow.co/2022/#section-most-popular-technologies-programming-scripting-and-markup-languages

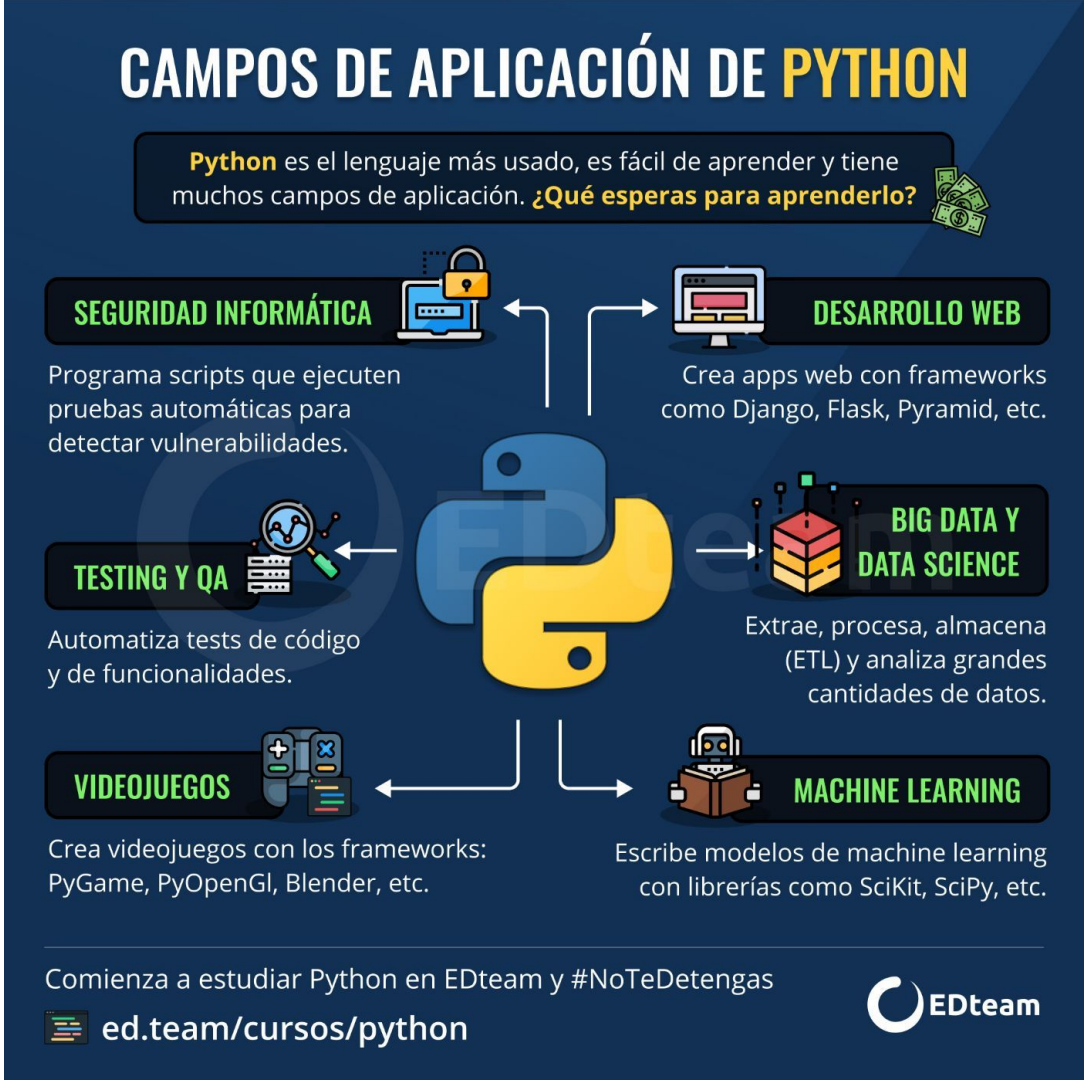
- Fácil de aprender.

  Tiene una curva de aprendizaje pequeña a diferencia de otros lenguajes.

- Lenguaje interpretado, no necesita ser compilado.

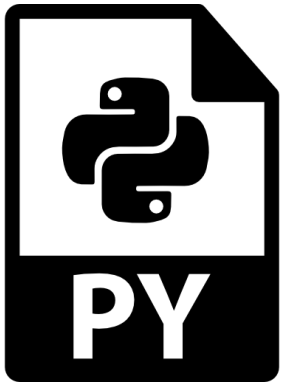
- **Multiparadigma.**
  - **Imperativo:**
    Sucesión de instrucciones/sentencias.
  - **Funcional:**
    Todos los elementos pueden entenderse como funciones y el código puede ejecutarse mediante llamadas de función secuenciales
  - **Programación Orientada a Objetos:**
    Consiste en modelos de objetos que representan elementos del problema a resolver.
    Estos objetos ayudan a separar los diferentes componentes de un programa y tienen características y funciones.

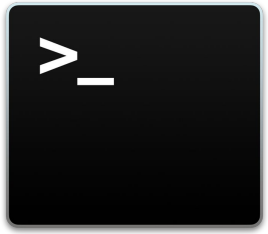
- Gran comunidad, bibliotecas y frameworks enfocados a diversas áreas.



- Es un lenguaje de programación dinámicamente/débilmente tipado.

¿Cómo ejecutar código de Python?

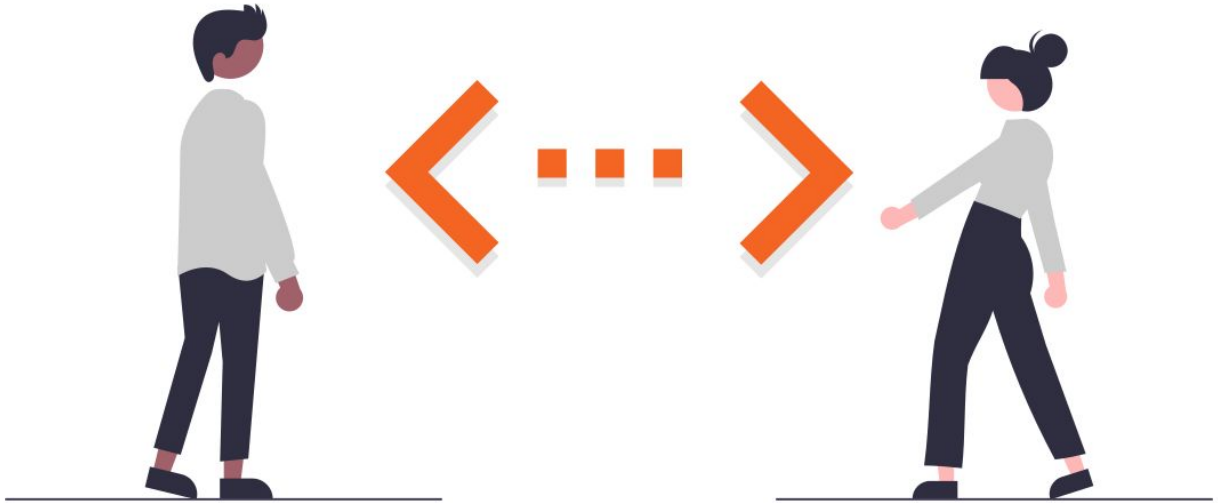





```
$ python script1.py
```

o

Sintaxis



¿Cómo se escribe código en Python?

- **Indentación****
- Declaración de Variables
- Tipos de datos
- Condicionales y Ciclos
- Funciones
- Clases

Declarar Variables

```
hello = "world!"
onehundred = 100
list_ = [1,2,3,4]
```

Tipos de datos

```
str_ = "String | Texto"
int_ = 10
float_ = 20.5
bool_ = True or False
list_ = [1,2,3,4,5]
tuple_ = (1,2,3,4,5)
dict_ = {"uno": 1, "dos": 2}
```

Condicionales

```
adult = True or False

if adult:
    pay_taxes()
```

Ciclos

```
# imprimir del 1 al 10
for x in range(10):
    print(x)
```

Funciones

```python
# función sin parámetros o retorno de valores
def diHola():
  print("Hello!")

diHola()  # llamada a la función, 'Hello!'

def multiplica(val1, val2):
  return val1 * val2

multiplica(3, 5)  # muestra 15 en la consola
```

Clases

```python
class Alumno:

    universidad = "UPQROO"

    def __init__(self, nombre):
        self.nombre = nombre

    def saludar(self):
        """Imprime un saludo en pantalla."""
        print(f"¡Hola, {self.nombre}!")


alumno = Alumno("Pablo")
alumno.saludar()
alumno.universidad
```

Importar Bibliotecas

```
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

Empresas que utilizan Python

NATIONAL GEOGRAPHIC

NETFLIX

WispHub

Gracias