

ÍNDICE

INTRODUCCIÓN	XIII
CAPÍTULO 1. PYTHON.....	1
INTRODUCCIÓN	1
Instalación de Python en Windows	2
Instalación de Python en Linux.....	5
Instalación de Python en MacOS.....	8
IDLE de Python	9
Cómo trabajar con archivos Python	11
Trabajando con el IDLE	13
Errores	14
Primeros pasos	14
Indentación	14
print.....	16
CAPÍTULO 2. VARIABLES.....	19
ALMACENANDO INFORMACIÓN.....	19
VARIABLES	19
Tipos de datos	21
Operadores de casting.....	23
Variables locales y globales	24
CAPÍTULO 3. OPERADORES.....	27
INTRODUCCIÓN A LOS OPERADORES	27
Operadores aritméticos.....	28
Operadores lógicos.....	29
Operadores de comparación	30

Operadores de asignación	31
Operadores de membresía	33
Operadores de identidad.....	33
Operadores Bitwise (bit a bit).....	34
CAPÍTULO 4. TUPLAS Y LISTAS	37
INTRODUCCIÓN A LAS TUPLAS Y LISTAS	37
Tuplas	37
Métodos de tuplas.....	40
Listas.....	41
Agregar o modificar el contenido de una lista	42
Eliminar elementos de una lista	44
Ordenar listas	45
Copiar listas	46
Indexar y contar.....	47
Elementos comunes a las tuplas y listas	47
CAPÍTULO 5. CONJUNTOS.....	51
CONJUNTOS PYTHON	51
Definir un conjunto.....	51
Insertar nuevos elementos en un conjunto.....	54
Eliminar elementos de un conjunto.....	55
Acceder a los elementos de un conjunto	56
Conjuntos y uniones	57
Conjuntos e intersecciones.....	58
Conjuntos y diferencias	59
Diferencias simétricas entre conjuntos	59
Conjuntos inmutables.....	60
CAPÍTULO 6. DICCIONARIOS	61
DICCIONARIOS PYTHON.....	61
Definición de un diccionario	61
Acceder a los elementos de un diccionario	63
Editar elementos de un diccionario	64
Agregar un elemento a un diccionario	65
Eliminar un elemento de un diccionario.....	65
Operaciones con diccionarios.....	67
Bucles en diccionarios	67
Comprensión de diccionarios	69
Chequeo de membresía en diccionarios.....	69
Contando claves de un diccionario	70
Copiar un diccionario.....	70

CAPÍTULO 7. ESTRUCTURAS DE CONTROL	73
INTRODUCCIÓN A LAS ESTRUCTURAS DE CONTROL	73
Estructuras de control condicionales	74
Estructuras de control iterativas	75
El bucle for.....	76
El bucle while.....	78
CAPÍTULO 8. FUNCIONES.....	81
INTRODUCCIÓN A LAS FUNCIONES	81
Funciones predefinidas	82
Funciones para el manejo de variables, listas, tuplas y diccionarios ..	82
Funciones para trabajar con texto.....	85
Funciones numéricas.....	85
Otras funciones útiles.....	87
Creación de funciones en Python.....	87
Pasar parámetros a funciones	89
Arbitrariedad de argumentos. *args y **kwargs.....	92
Funciones Lambda.....	95
Funciones recursivas	98
CAPÍTULO 9. MÓDULOS	101
INTRODUCCIÓN A LOS MÓDULOS	101
Creación e importación de un módulo	101
Ruta de búsqueda del módulo de Python	104
Recarga de un módulo.....	105
Paquete Python	106
La función integrada dir()	106
USO DE MÓDULOS	107
Módulo datetime.....	109
Módulo math.....	112
Módulo random.....	113
CAPÍTULO 10. CLASES, OBJETOS Y MÉTODOS	115
INTRODUCCIÓN A LA POO	115
Las clases	116
Atributos, métodos y funciones de un objeto	117
Los métodos	119
Herencia	121
Herencia múltiple	122
Herencia multinivel	123
Encapsulación y polimorfismo en Python.....	124
Encapsulación en Python.....	124
Polimorfismo en Python	125

CAPÍTULO 11. EXCEPCIONES PYTHON	127
INTRODUCCIÓN A LAS EXCEPCIONES	127
try y except	128
CAPÍTULO 12. TRABAJANDO CON ARCHIVOS.....	133
GESTIÓN DE ARCHIVOS CON PYTHON	133
Archivos con Python	133
Cómo acceder a los archivos	134
Trabajar con archivos	135
Trabajar con directorios	139
CAPÍTULO 13. EXPRESIONES REGULARES.....	141
INTRODUCCIÓN A LAS REGEX.....	141
Componentes de las expresiones regulares	141
Metacaracteres	147
Secuencias especiales.....	149
Rangos	151
CAPÍTULO 14. PYTHON MYSQL.....	153
INTRODUCCIÓN A MYSQL.....	153
Base de datos MySQL	154
Driver MySQL.....	155
Conexión a MySQL.....	157
Creación de una base de datos.....	158
Creación de tablas	159
Entrada de datos en las tablas.....	163
Selección de registros.....	166
Eliminación de registros, tablas y bases de datos.....	171
Manejo de excepciones en MySQL.....	173
CAPÍTULO 15. PYTHON SQLITE3.....	177
INTRODUCCIÓN A SQLITE3	177
Conectando con SQLite3	177
Creación de tablas	178
Insertar y seleccionar datos de una tabla.....	180
Otras operaciones con datos.....	182
Ejemplo base de datos de países.....	184
CAPÍTULO 16. FRAMEWORKS PYTHON	191
INTRODUCCIÓN A FRAMEWORKS	191
Django	193
Flask.....	193
web2py.....	194
¿Y ahora qué?.....	195

CAPÍTULO 17. FLASK	197
INTRODUCCIÓN A FLASK	197
Creación de un entorno virtual en Python	198
Instalación de Flask	200
Primeros pasos con Flask.....	201
Plantillas	205
Variables.....	208
Estructuras condicionales.....	210
Gestión de bases de datos.....	213
APÉNDICE A	223
GUÍA DE CONSULTA RÁPIDA DE PYTHON	223
Generalidades.....	223
Tipos de datos de Python	224
Operadores aritméticos.....	224
Operadores lógicos.....	225
Operadores de comparación	225
Operadores de asignación	226
Operadores de membresía.....	227
Operadores de identidad.....	228
Operadores Bitwise (bit a bit).....	228
Palabras reservadas.....	229
Secuencias	229
Diccionarios (dict).....	229
Conjuntos	229
Estructuras condicionales.....	230
Funciones	231
Clases.....	232
Módulos.....	233
APÉNDICE B	235
VISUAL STUDIO CODE	235
Instalación	235
Configuración de VS Code	237
Extensiones.....	238
Primeros pasos	240
IntelliSense	241
Detección de errores	242
Atajos de teclado.....	243
APÉNDICE C	245
INSTALACIÓN DE MySQL	245
Instalación de MySQL en Linux.....	245

Instalación de MySQL en MacOS y Windows.....	248
Instalación en MacOS	250
Instalación en Windows.....	251
ÍNDICE ANALÍTICO.....	255

INTRODUCCIÓN

¿QUE ES PYTHON?

Esta obra está dirigida a todas aquellas personas que quieran aprender a programar con Python. El libro es ante todo un texto práctico que explica mediante numerosos ejemplos los principales elementos del lenguaje.

Python es un lenguaje de programación muy popular. Fue creado por Guido Van Rossum sobre los 90. Actualmente, la evolución del lenguaje Python es gestionada por la **Python Software Foundation**, una sociedad sin ánimo de lucro que se dedica a dar difusión al lenguaje y apoyar su evolución.

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 la programación **orientación a objetos**, programación **imperativa**, programación **funcional** y es **dinámicamente tipado**. Finalmente, estamos ante un lenguaje **multiplataforma** y libre.

Vamos a ver con detalle los términos marcados en negrita:

- **Interpretado.** El código no necesita compilación para ser ejecutado. El intérprete lo va traduciendo a medida que es necesario. Una de las grandes ventajas de este tipo de lenguaje es que, al no necesitar compilación, ahorra muchas horas de desarrollo y prueba de una aplicación.

- **Multiparadigma.** Un paradigma de programación es un estilo de programación de software. Existen diferentes formas de diseñar un lenguaje de programación y varios modos de trabajar para obtener los resultados que necesitan los programadores. Los paradigmas son un conjunto de métodos sistemáticos aplicables en todos los niveles del diseño de programas para resolver problemas computacionales.

Python admite:

- **Imperativo.** La programación imperativa consiste en una sucesión de instrucciones o conjunto de sentencias. El desarrollador escribe con su código paso por paso todo lo que hará el programa.
 - **Funcional.** La programación funcional se basa en el uso de funciones matemáticas que permite la variación del programa mediante la mutación de variables.
 - **Orientado a Objetos.** La programación orientada a objetos se basa en la construcción de elementos (objetos) que agrupan características y propiedades afines a dicho objeto.
- **Tipado dinámico.** Lenguaje en que sus variables pueden tomar valores de distinto tipo. Las variables son declaradas por su contenido y no por su contenedor.
 - **Multiplataforma.** Lenguaje que puede ejecutarse en diferentes sistemas operativos (Windows, Linux, MacOS, etc.).

Python se utiliza para multitud de objetivos, pero principalmente destacaríamos:

- Desarrollo web del lado del servidor.
- Desarrollo de software.
- Matemáticas.
- Secuencias de comandos del sistema.
- Ciencia de datos.

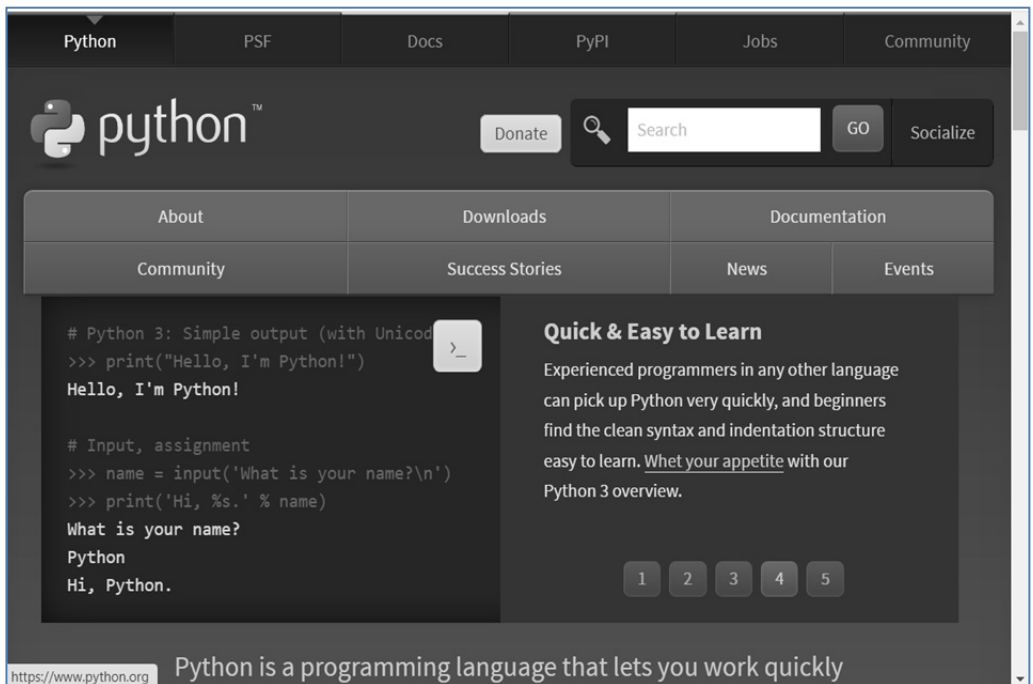
Python es un lenguaje que se ha hecho muy popular estos últimos años por diferentes motivos, pero sobre todo destacaríamos:

- Python se desarrolla bajo una licencia de Open Source, por lo que se puede usar y distribuir libremente incluso para usos comerciales.
- Funciona en diferentes plataformas (Windows, Mac, Linux, etc.).
- Tiene una sintaxis muy simple y similar al idioma inglés.

- Su sintaxis permite escribir programas con menos líneas que con otros lenguajes de programación.
- Los programas se ejecutan sin necesidad de ser compilados.
- Usa nuevas líneas para completar un comando a diferencia de otros lenguajes de programación que usan punto y coma o paréntesis.

En el momento de la redacción de este curso práctico de formación, la versión actual de Python es la 3.10.1. Podemos conocer en todo momento las novedades del lenguaje desde la página principal de Python:

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



Pantalla de inicio de la página oficial de Python

Por todos los motivos referenciados antes, Python es actualmente uno de los lenguajes de programación más populares.

Su éxito viene determinado por su potencia, velocidad e integración con otros lenguajes de programación. Finalmente, la integración con otros lenguajes y el poder ejecutarse en prácticamente cualquier plataforma convierten a Python en la herramienta de programación perfecta para los desarrolladores de software actuales.

Un ejemplo del potencial de este lenguaje y su popularidad es saber que, por ejemplo, Google o plataformas como Netflix o Youtube, han sido desarrolladas con Python.

Cómo trabajar con este libro

Para aprovechar al máximo este curso, se recomienda seguir todos los capítulos por orden y realizar los distintos ejercicios que van apareciendo. Probar los ejemplos, modificarlos y ver cómo actúa el programa es una forma estupenda de empezar a familiarizarse con Python.

Como hemos indicado anteriormente, una de las grandes ventajas de Python es que no necesita ningún software especial para programar. Basta un simple editor de texto para empezar a trabajar. Sin embargo, el hecho de que podamos usar ese editor de texto para escribir código, no quiere decir que sea la única manera.

Para trabajar óptimamente se recomienda el uso de un entorno de desarrollo integrado llamado **IDE (*Integrated Development Environment*)**, que no es más que una aplicación que proporciona los medios para facilitar al desarrollador su trabajo.

El IDE en sí mismo es un software que consta de herramientas que se utilizan para desarrollar software y probarlo. Proporciona un entorno de desarrollo donde todas las herramientas están disponibles en una única interfaz gráfica de usuario (GUI) fácil de usar. Principalmente un IDE dispone de un editor de código, automatización de la entrada de programa y depurador de programas.

En este libro, nos vamos a centrar en IDE de la distribución de Python: **IDLE**, entorno de aprendizaje y desarrollo integrado que permite editar y ejecutar programas en Python.

IDLE es también un entorno interactivo con el que podemos ejecutar instrucciones sueltas de Python.

Como se ha comentado anteriormente, Python es multiplataforma:

- En Windows, IDLE se distribuye junto con el intérprete de Python, es decir, al instalar Python en Windows también se instala IDLE.
- En Linux y MacOS, IDLE se distribuye como una aplicación separada que se puede instalar desde los repositorios de cada distribución.

Consideramos que el IDLE de la distribución de Python es la mejor opción para seguir con el aprendizaje del curso práctico de este libro, ahora bien, otros editores como **Visual Studio Code (Vs code)** pueden ser, una vez conozcamos el lenguaje de programación, mucho más útiles al disponer de módulos que permiten mejorar el desarrollo de aplicación y sobre todo, la disponibilidad de extensiones para otros lenguajes como C ++, C #, PHP, etc.

El **apéndice B** de este libro está dedicado a esta aplicación con la que el lector podrá adquirir los conocimientos básicos para empezar a utilizarla.

Convenciones utilizadas en este libro

En este libro se utilizan una serie de convencionalismos que permiten hacer más clara la lectura de la obra y el seguimiento del curso.

Veamos a continuación cuáles son:



Cuando aparezca este icono en algún punto del texto, es importante prestar atención. Encontraremos una advertencia que evitará futuros problemas o quebraderos de cabeza.



Este icono indica que nos encontramos antes una nota curiosa, idea o comentario que puede resultar útil al lector. Se trata de información importante para tener en cuenta.

Los códigos de ejemplo que van a ir saliendo a lo largo del curso aparecerán destacados de varias formas.

Cuando estemos representando información desde **el entorno interactivo del Shell de Python** el código aparecerá de esta forma:

```
>>> # Ejemplo de código Python

>>> a= '1 '
>>> b= '2 '
>>> print (a+b)
```

12

El símbolo >>> no hay que escribirlo, sirve solamente para representar dónde escribimos nosotros el código y lo que responde el ordenador (sin >>>).

No obstante, la **mayoría de los ejemplos de este libro** serán “miniprogramas” escritos con el editor IDLE, por lo que no aparecerán los símbolos del *prompt* y el resultado del programa, en caso de que sea necesario mostrarlo, aparecerá debajo sin el fondo gris.

```
nombre = 'Carlos '
if nombre == 'Carlos!':
    print('Carlos ')
else:
    print('No eres Carlos ')
```

Carlos

EL CÓDIGO CONTINÚA EN LA PÁGINA SIGUIENTE



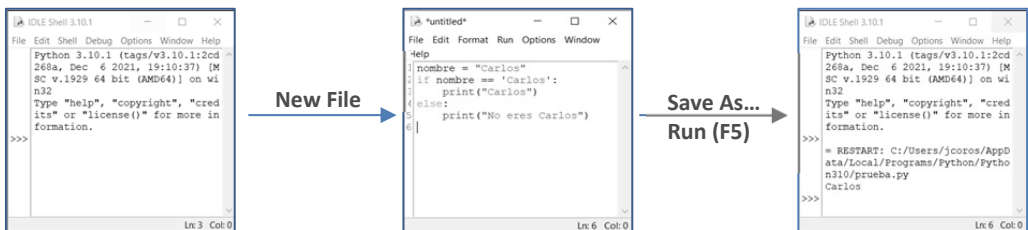
Cómo ejecutar los programas de ejemplo

Los códigos que aparezcan en el libro sin el *prompt* del Shell son programas que deben ser ejecutados para obtener el correspondiente resultado.

IDLE de Python

En el siguiente capítulo se explica con detalle cómo trabajar con el IDLE de Python, pero destacamos aquí que el proceso para testear los programas que aparecen en este curso de formación siempre será el mismo.

1. Abrir el IDLE de Python.
2. Abrir una ventana nueva (*New File*) donde escribiremos el código Python.
3. Guardar el programa asignando un nombre, por ejemplo, **prueba.py**.
El editor asigna automáticamente la extensión `.py` al programa.
4. Seleccionar la opción Run o (F5) para ejecutar el código.



Terminal

Si los programas de Python los vamos a ejecutar desde la terminal de nuestro sistema operativo, deberemos abrir dicha terminal y ejecutar el programa escribiendo su nombre junto con la extensión de Python `.py`. Para los siguientes ejemplos, el programa que se va a ejecutar se llama *prueba.py*.

En Windows, abriremos la aplicación Símbolo de sistema (CMD) donde deberemos escribir el nombre del programa directamente, *py* y el nombre del programa, o bien, *python* y el nombre del programa. Podemos verlo con detalle en la siguiente imagen.

```
D:\PythonCode>prueba.py
Hola Mundo

D:\PythonCode>py prueba.py
Hola Mundo

D:\PythonCode>python prueba.py
Hola Mundo
```

En la terminal de MacOS tendremos que escribir *python* y el nombre del programa para usar el intérprete por defecto, o bien, en nuestro caso, la versión de Python instalada en el sistema (ver imagen inferior).

```
Esitorio --zsh-- 83x13
[carlosys@MacBook-Carlosys desktop % python3.10 prueba.py
Hola Mundo
carlosys@MacBook-Carlosys desktop % █
```

En Linux, abriremos la terminal correspondiente y escribiremos *python3* y a continuación el nombre del programa.

```
carlinux@PCCarlinux: ~/Escritorio
carlinux@PCCarlinux:~/Escritorio$ python3 prueba.py
Hola Mundo
carlinux@PCCarlinux:~/Escritorio$ █
```

Agradecimientos

Escribir un libro implica consumir parte del escaso y valioso tiempo libre disponible para la familia; por este motivo, deseo reconocer una vez más la comprensión y el apoyo incondicional de mi esposa, Nieves, y de mis hijas, Lydia y Sonia, por haber aguantado una vez más mi encierro en el estudio mientras escribía este libro.

Agradecer nuevamente a todos mis lectores de España e Hispanoamérica las felicitaciones recibidas de mis otros libros de programación web. Sin duda, esos reconocimientos animan a afrontar el reto de escribir un libro inédito en mí, sobre el lenguaje de programación Python.

Finalmente, y como no puede ser de otra forma, dar las gracias a todo el equipo de **RC Libros** y en especial a José Luis Ramírez, por seguir confiando en mi trabajo después de tantos años.

Gracias a todos.

Juan Carlos Orós

1 PYTHON

INTRODUCCIÓN

El primer paso para aprender a programar en Python es sin duda, disponer del intérprete en nuestro sistema.

Los sistemas operativos Linux y MacOS ya lo traen por defecto, aunque probablemente será necesaria una actualización a la última versión de Python ya que la **Python Software Foundation** es muy activa y está permanentemente revisando y actualizando las distribuciones de Python.

Por el contrario, en Windows sí que es necesaria su instalación desde cero ya que no viene instalado por defecto.

Vamos a ver a continuación cómo instalar o actualizar Python en los sistemas operativos Windows, Linux y MacOS.



Los procedimientos que se describen a continuación y las capturas de pantalla que se presentan están basadas en la distribución actual que ofrece Python en su portal web (la versión 3.10.1).

Dichas imágenes pueden variar en futuras versiones del instalador, no obstante, los procedimientos y objetivos serán casi con total seguridad, similares a los descritos.

Instalación de Python en Windows

Para instalar Python en el sistema operativo Windows necesitamos descargar el paquete de instalación desde la web de Python.

Deberemos acceder a la opción de descargar de la web oficial de Python y seleccionar el archivo de instalación que mejor se adapte a la arquitectura del hardware de nuestro ordenador.



Normalmente, el propio motor web del navegador pasa información a la web y esta presenta en pantalla la versión recomendada para la descarga.

En el momento de la redacción de este manual, la versión disponible es la 3.10.1:

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

The screenshot shows the Python.org website with the following elements:

- Navigation menu: Python, PSF, Docs, PyPI, Jobs, Community
- Secondary menu: About, Downloads, Documentation, Community, Success Stories, News, Events
- Main heading: **Download the latest version for Windows**
- Primary button: **Download Python 3.10.1** (circled in red)
- Text: Looking for Python with a different OS? Python for Windows, Linux/LINUX, macOS, Other
- Text: Want to help test development versions of Python? Prereleases, Docker Images
- Text: Looking for Python 2.7? See below for specific releases
- Section: Active Python Releases
- Text: For more information visit the Python Developer's Guide.
- Table of Active Python Releases:

Python version	Maintenance status	First released	End of support	Release schedule
3.10	bugfix	2021-10-04	2026-10	PEP 619
3.9	bugfix	2020-10-05	2025-10	PEP 596
3.8	security	2019-10-14	2024-10	PEP 569

Figura 1.1. Download de Python con la distribución de Windows recomendada para su descarga en relación con el sistema operativo del equipo local

Una vez descargado el archivo, haremos doble clic sobre él para iniciar el asistente de instalación de Python. El propio instalador nos irá guiando con las distintas fases de instalación.

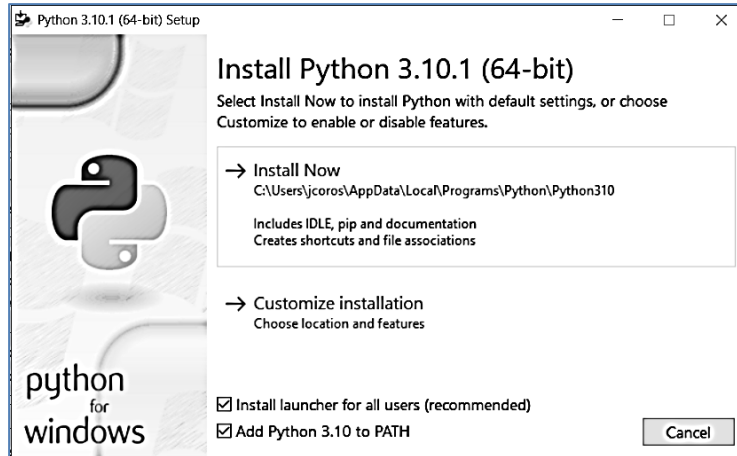


Figura 1.2. Ventana principal de instalación de Python

En la pantalla principal de instalación, deberemos seleccionar la opción *Install Now* para instalar en el directorio por defecto del sistema, o bien, *Customize installation* para seleccionar un directorio diferente. En dicha ventana, también seleccionaremos las opciones *Install launcher for all users* y *Add Python 3.10 to PATH*.

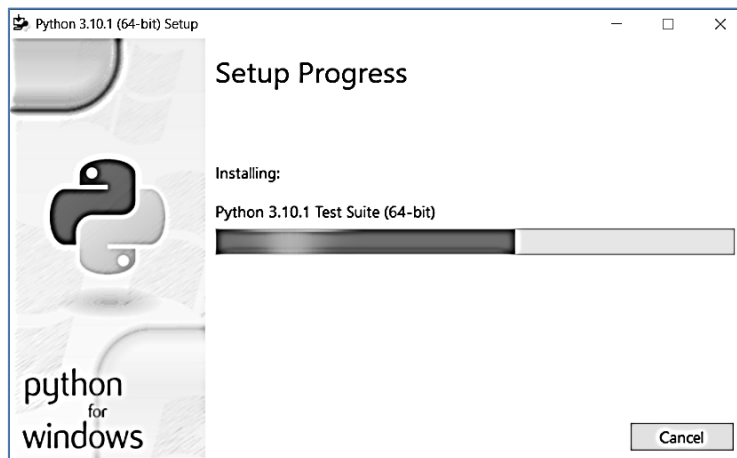
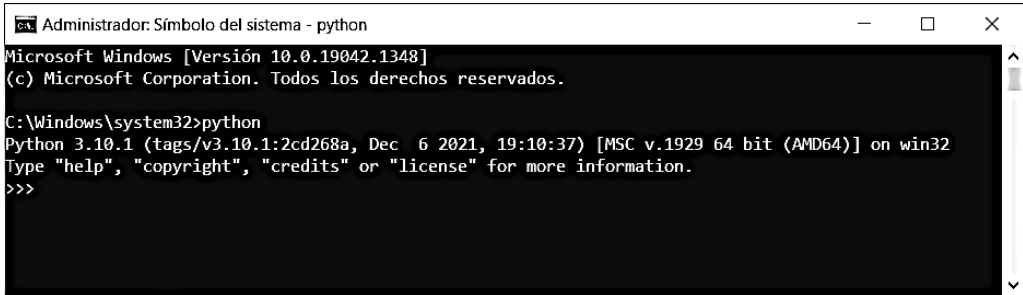


Figura 1.3. Progreso de la instalación de Python

Cuando finalice el proceso de instalación aparecerá una ventana indicando **Setup was successful**, mensaje que indica que ya dispondremos del intérprete de Python en el sistema y el Shell IDLE que usaremos para el desarrollo de este curso.

Solo queda verificar que ambos puntos funcionan correctamente.

El primer paso es ver si desde el terminal de comandos de Windows podemos trabajar con Python. Para ello, abriremos la terminal de Símbolo de sistema y escribiremos en ella la palabra “python”. Si todo está Ok, la terminal devolverá un mensaje similar al siguiente:



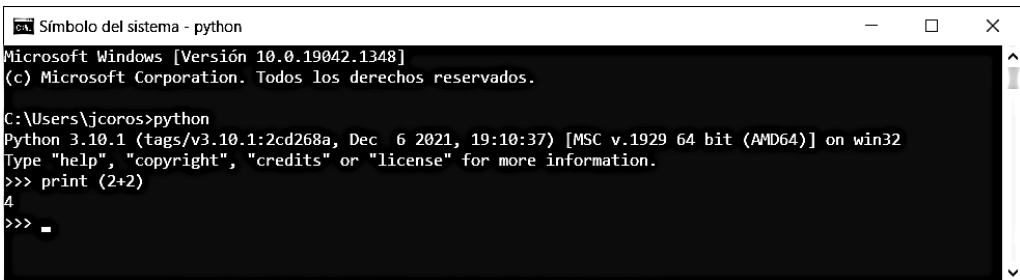
```
Administrador: Símbolo del sistema - python
Microsoft Windows [Versión 10.0.19042.1348]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Windows\system32>python
Python 3.10.1 (tags/v3.10.1:2cd268a, Dec 6 2021, 19:10:37) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Podríamos incluso probar con un sencillo código para tener la certeza de que realmente, aparte de estar instalado, funciona correctamente. Para ello, escribiremos en la terminal:

```
print (2+2)
```

El sistema devolverá el resultado de la operación.

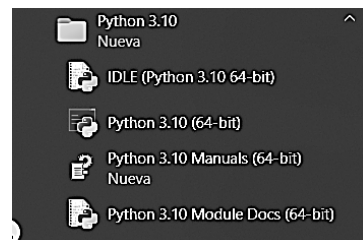


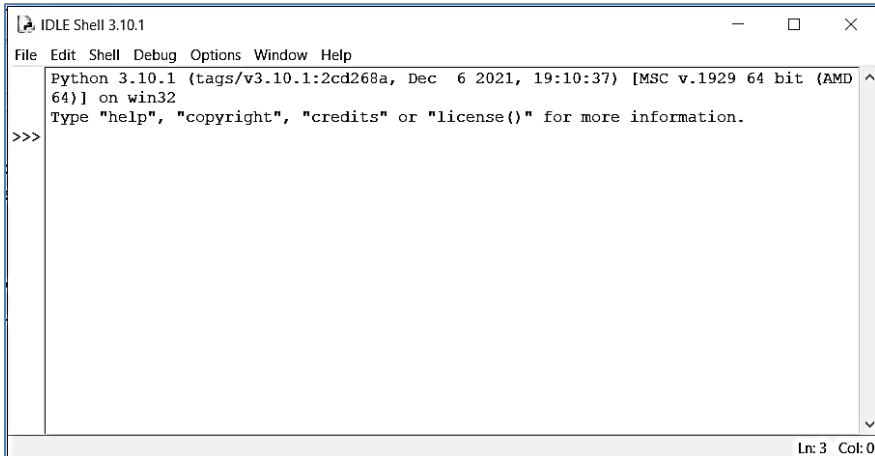
```
Símbolo del sistema - python
Microsoft Windows [Versión 10.0.19042.1348]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\jcoros>python
Python 3.10.1 (tags/v3.10.1:2cd268a, Dec 6 2021, 19:10:37) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print (2+2)
4
>>> _
```

Para salir del intérprete de Python bastará con escribir **quit()** y la terminal volverá al *prompt* de directorios de Windows.

El siguiente punto para verificar es comprobar si el IDLE de Python se ha instalado correctamente. Para ello, solo tendremos que ejecutar la aplicación IDLE instalada y ver si se abre la ventana del Shell de Python junto con los distintos menús de trabajo que nos ofrece el entorno de desarrollo.





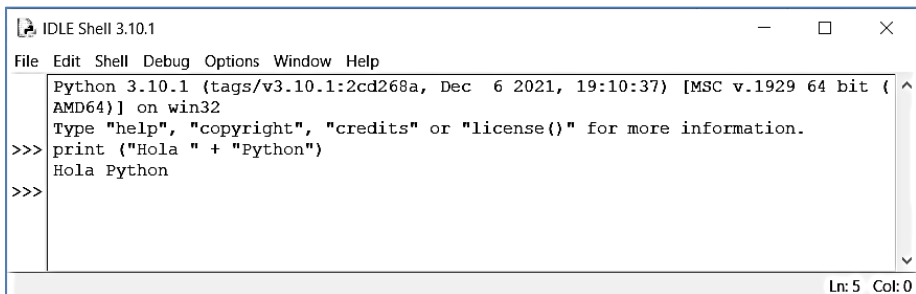
```

Python 3.10.1 (tags/v3.10.1:2cd268a, Dec 6 2021, 19:10:37) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>

```

Figura 1.4. Ventana del IDLE de Python

Al igual que hicimos con la terminal del sistema de Windows, podemos probar su funcionalidad escribiendo un pequeño código y ver si responde adecuadamente.



```

Python 3.10.1 (tags/v3.10.1:2cd268a, Dec 6 2021, 19:10:37) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print ("Hola " + "Python")
Hola Python
>>>

```

Figura 1.5. Ventana del IDLE donde vemos cómo se ejecuta un pequeño código

Instalación de Python en Linux

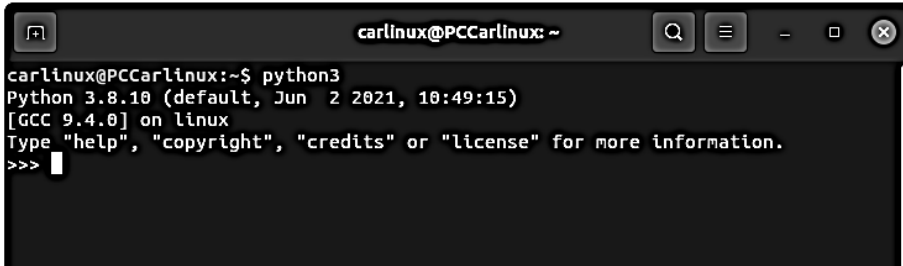
Como se ha comentado anteriormente, la mayoría de las distribuciones de Linux llevan instalado Python por defecto independientemente de que para el IDLE exista el correspondiente paquete de instalación aparte.



Se describen a continuación los pasos para verificar la instalación de Python en Linux bajo la distribución de Ubuntu, por lo que pueden haber pequeñas variaciones en los pasos que se explican para otras distribuciones de Linux.

El primer paso que deberemos realizar para trabajar con Python en Linux es verificar si en nuestra distribución está instalado Python y la versión que tenemos instalada.

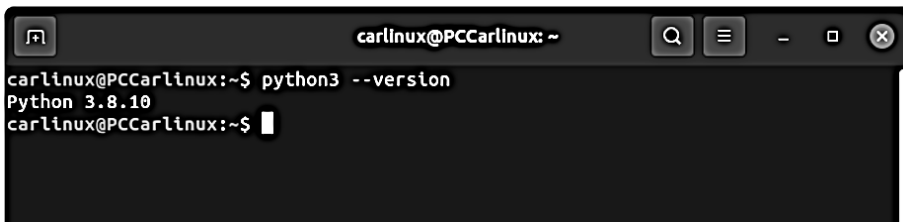
Para ello, abriremos la terminal de Linux y escribiremos *python3* para entrar en el *Shell* y ver qué versión está instalada en el sistema.



```
carlinux@PCCarlinux: ~  
carlinux@PCCarlinux:~$ python3  
Python 3.8.10 (default, Jun 2 2021, 10:49:15)  
[GCC 9.4.0] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> |
```

También podemos ejecutar el siguiente comando:

```
python3 --version
```



```
carlinux@PCCarlinux: ~  
carlinux@PCCarlinux:~$ python3 --version  
Python 3.8.10  
carlinux@PCCarlinux:~$ |
```

En ambos casos, vemos que la versión de Python instalada es la 3.8.10. Esta versión permite seguir el curso de este libro sin ningún tipo de problema, sin embargo, vamos a ver cómo actualizar Python a la última distribución disponible.

Antes que actualizar Python es importante asegurarse de que el sistema operativo y todos sus paquetes están actualizados.

Solo una vez confirmada la correcta actualización del sistema Linux, deberemos seguir con la actualización de Python.

Para ello, ejecutaremos la siguiente línea de código:

```
sudo apt update && sudo apt upgrade -y
```