

CAPÍTULO 1. INTRODUCCIÓN A LA PROGRAMACIÓN Y A PYTHON

1. ¿Qué es programar?

Vivimos rodeados de tecnología. Utilizamos teléfonos móviles, ordenadores, videojuegos, aplicaciones de mensajería, asistentes virtuales y páginas web. Todos estos sistemas funcionan gracias a programas informáticos.

Programar consiste en diseñar y escribir instrucciones para que un ordenador realice una tarea determinada.

Un ordenador puede realizar millones de operaciones por segundo, pero no puede tomar decisiones por sí mismo. Necesita que una persona le indique exactamente qué debe hacer y en qué orden debe hacerlo.

Por ejemplo, si queremos que un ordenador calcule la media de tres notas, tendremos que indicarle:

1. Qué notas debe utilizar.
2. Cómo calcular la media.
3. Cómo mostrar el resultado.

El conjunto de instrucciones que permiten resolver un problema recibe el nombre de programa.

2. Algoritmos, programas y pseudocódigo

Cuando una persona se enfrenta a un problema, normalmente sigue una serie de pasos para resolverlo.

Por ejemplo, si queremos calcular la media de tres notas, podríamos hacer lo siguiente:

1. Anotar las tres notas.
2. Sumarlas.
3. Dividir el resultado entre tres.
4. Mostrar la media obtenida.

Esta secuencia ordenada de pasos recibe el nombre de **algoritmo**.

¿Qué es un algoritmo?

Un algoritmo es un conjunto finito y ordenado de instrucciones que permiten resolver un problema o realizar una tarea.

Los algoritmos no son exclusivos de la informática. También utilizamos algoritmos en nuestra vida cotidiana.

Ejemplo: preparar un bocadillo

Algoritmo:

1. Cortar el pan.
2. Introducir el relleno.
3. Cerrar el bocadillo.
4. Servir.

Ejemplo: calcular la media de tres notas

Algoritmo:

1. Leer las tres notas.
 2. Sumarlas.
 3. Dividir la suma entre tres.
 4. Mostrar el resultado.
-

Diferencia entre algoritmo y programa

Es importante distinguir estos dos conceptos.

Algoritmo

Es la descripción de los pasos necesarios para resolver un problema.

Programa

Es la implementación de un algoritmo utilizando un lenguaje de programación.

Por ejemplo:

Algoritmo

1. Mostrar un saludo.

Programa en Python

```
print("Hola")
```

El algoritmo es la idea. El programa es la forma de comunicar esa idea al ordenador.

¿Qué es el pseudocódigo?

Antes de escribir un programa, los programadores suelen diseñar el algoritmo utilizando una representación sencilla denominada **pseudocódigo**.

El pseudocódigo utiliza palabras similares al lenguaje natural y evita detalles propios de un lenguaje concreto.

Por ejemplo:

Escribir "Hola"

Otro ejemplo:

Problema: calcular el área de un cuadrado.

Pseudocódigo

```
Leer lado  
area ← lado × lado  
Escribir area
```

Observa que el pseudocódigo no pertenece a ningún lenguaje de programación concreto.

Su objetivo es describir claramente los pasos necesarios para resolver un problema.

¿Por qué utilizaremos pseudocódigo?

A lo largo de este manual seguiremos siempre la misma estrategia:

1. Comprender el problema.
2. Diseñar un algoritmo.
3. Escribir el pseudocódigo.
4. Implementar la solución en Python.

Este método ayuda a desarrollar el pensamiento lógico y facilita la resolución de problemas más complejos.

3. ¿Qué es un lenguaje de programación?

Los ordenadores no entienden el lenguaje humano.

Para comunicarnos con ellos utilizamos lenguajes de programación.

Un lenguaje de programación es un conjunto de reglas y símbolos que permite escribir programas.

Existen muchos lenguajes de programación:

- Python
- Java
- C++
- JavaScript
- C#
- PHP

Cada uno tiene características diferentes y se utiliza en ámbitos distintos.

En este curso utilizaremos Python por su sencillez y por ser uno de los lenguajes más utilizados en la actualidad.

4. ¿Por qué aprender Python?

Python fue creado por Guido van Rossum en 1991.

Actualmente se utiliza en numerosos campos:

- Inteligencia artificial.
- Ciencia de datos.
- Investigación científica.
- Desarrollo web.
- Automatización de tareas.
- Robótica.

Python es un lenguaje ideal para iniciarse en la programación porque:

- Tiene una sintaxis sencilla.
 - Es fácil de leer.
 - Permite centrarse en la resolución de problemas.
 - Cuenta con una enorme comunidad de usuarios.
-

5. Nuestro entorno de programación

Para realizar las actividades de este manual utilizaremos un entorno de programación online.

Vamos a utilizar el editor on-line :

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

Esto nos permitirá escribir y ejecutar programas desde cualquier ordenador con conexión a Internet sin necesidad de instalar software adicional.

La mayoría de los entornos online tienen una estructura similar.

El editor de código

Es la zona donde escribiremos nuestros programas.

Por ejemplo:

```
print("Hola mundo")
```

La consola de salida

Es la zona donde aparecen los resultados obtenidos al ejecutar el programa.

Si ejecutamos el programa anterior veremos:

Hola mundo

Cómo ejecutar un programa

Para ejecutar un programa seguiremos siempre estos pasos:

1. Escribir el código en el editor.
2. Pulsar el botón **Run, Ejecutar** o ►.
3. Observar el resultado en la consola.

Normas básicas

Durante este curso debemos recordar que:

- Python distingue entre mayúsculas y minúsculas.
 - Cada instrucción suele escribirse en una línea.
 - Los errores son normales y forman parte del aprendizaje.
 - Leer los mensajes de error ayuda a encontrar problemas en el código.
-

6. Nuestro primer programa

Tradicionalmente, el primer programa que se escribe en cualquier lenguaje de programación muestra el mensaje “Hola mundo”.

Escribe el siguiente código:

```
print("Hola mundo")
```

Al ejecutarlo aparecerá:

```
Hola mundo
```

Aunque se trata de un programa muy sencillo, ya contiene varios elementos importantes.

7. La función print()

La función `print()` permite mostrar información en pantalla.

Su sintaxis básica es:

```
print("mensaje")
```

Ejemplo:

```
print("Bienvenidos al curso de Python")
```

Resultado:

```
Bienvenidos al curso de Python
```

Podemos utilizar varias instrucciones `print()` dentro del mismo programa.

```
print("Nombre: Ana")  
print("Curso: 2º Bachillerato")  
print("Asignatura: TIC")
```

Resultado:

```
Nombre: Ana  
Curso: 2º Bachillerato  
Asignatura: TIC
```

Cada llamada a `print()` genera una nueva línea en la consola.

8. Analizando nuestro primer programa

Observemos nuevamente el código:

```
print("Hola mundo")
```

En esta instrucción aparecen varios elementos:

- `print` es una función.
 - Los paréntesis indican que estamos utilizando esa función.
 - Las comillas delimitan el texto que queremos mostrar.
 - El texto entre comillas recibe el nombre de cadena de caracteres.
-

9. Comentarios y documentación del código

Cuando los programas crecen, resulta útil añadir explicaciones para comprender mejor qué hace cada parte del código.

Estas explicaciones reciben el nombre de comentarios o documentación.

Comentarios de una línea

La forma más habitual de escribir comentarios en Python es utilizando el símbolo `#`.

```
# Mostrar un mensaje de bienvenida  
print("Hola")
```

Todo el texto situado después del símbolo `#` será ignorado por Python.

También podemos escribir comentarios al final de una instrucción.

```
print("Hola") # Mensaje de bienvenida
```

Comentarios de varias líneas

Si necesitamos escribir una explicación larga, podemos utilizar varias líneas que comiencen por `#`.

```
# Programa de ejemplo  
# Creado por un alumno de TIC  
# Curso 2026-2027  
  
print("Hola")
```

Uso de las tres comillas

Python permite escribir texto entre tres comillas dobles ("""") o tres comillas simples (''').

```
"""
Programa de ejemplo
Creado por un alumno
Curso 2026-2027
"""
```

o bien:

```
'''
Programa de ejemplo
Creado por un alumno
Curso 2026-2027
'''
```

Aunque muchos programadores utilizan esta técnica para escribir explicaciones largas, en realidad no son comentarios propiamente dichos.

Python interpreta este contenido como una **cadena de texto multilínea**.

Sin embargo, cuando dicha cadena no se utiliza para nada, suele emplearse como una forma cómoda de documentar programas.

Por ejemplo:

```
"""
Programa que muestra un saludo.
Primer ejemplo del curso.
"""
```

```
print("Hola")
```

Como norma general:

- Utilizaremos # para escribir comentarios.
- Utilizaremos las tres comillas cuando queramos añadir documentación más extensa.
- Más adelante veremos que las tres comillas se utilizan también para documentar funciones y programas profesionales.

10. Errores frecuentes

Cometer errores es una parte normal del aprendizaje.

Los errores nos ayudan a comprender mejor el funcionamiento del lenguaje.

Error 1: olvidar las comillas

Código incorrecto:

```
print(Hola)
```

Código correcto:

```
print("Hola")
```

Error 2: escribir mal una instrucción

Código incorrecto:

```
prin("Hola")
```

Código correcto:

```
print("Hola")
```

Error 3: olvidar un paréntesis

Código incorrecto:

```
print("Hola"
```

Código correcto:

```
print("Hola")
```

11. Ejemplos resueltos

Ejemplo 1. Mostrar un nombre

```
print("Carlos")
```

Ejemplo 2. Mostrar información personal

```
print("Nombre: Carlos")  
print("Edad: 17 años")  
print("Curso: 2º Bachillerato")
```

Ejemplo 3. Utilizar comentarios

Programa de presentación

```
print("Nombre: Carlos")  
print("Curso: 2º Bachillerato")
```

Ejemplo 4. Utilizar documentación con tres comillas

"""

Programa de presentación

Realizado para la asignatura TIC

"""

```
print("Nombre: Carlos")  
print("Curso: 2º Bachillerato")
```

Actividades propuestas

Actividades básicas

1. Escribe un programa que muestre tu nombre.
 2. Escribe un programa que muestre tu edad.
 3. Escribe un programa que muestre el nombre de tu instituto.
 4. Escribe un programa que muestre tu ciudad.
-

Actividades de práctica

5. Muestra tu nombre y apellidos.
 6. Muestra tres mensajes en líneas diferentes.
 7. Escribe una frase relacionada con la tecnología.
 8. Escribe una frase relacionada con la inteligencia artificial.
 9. Crea un programa que utilice cinco instrucciones `print()`.
 10. Añade comentarios utilizando `#`.
 11. Añade una descripción inicial utilizando tres comillas.
-

Actividades de algoritmia

12. Escribe el algoritmo para preparar un bocadillo.

13. Escribe el algoritmo para calcular la media de tres notas.
 14. Escribe el pseudocódigo para mostrar un mensaje de bienvenida.
 15. Escribe el pseudocódigo para calcular el área de un cuadrado.
-

Actividades de investigación

16. Busca información sobre Guido van Rossum y redacta un resumen de cinco líneas.
 17. Investiga tres aplicaciones reales donde se utilice Python.
 18. Busca qué empresas utilizan Python en sus productos o servicios.
-

Reto final

Diseña un programa que muestre una ficha personal con el siguiente formato:

```
-----  
FICHA PERSONAL  
-----  
Nombre: Ana García  
Edad: 17 años  
Curso: 2º Bachillerato  
Aficiones: música, deporte y lectura  
-----
```

Añade al principio del programa una breve documentación utilizando tres comillas que explique qué hace el programa.

Antes de escribir el código:

1. Diseña el algoritmo.
 2. Escribe el pseudocódigo.
 3. Implementa la solución en Python.
-

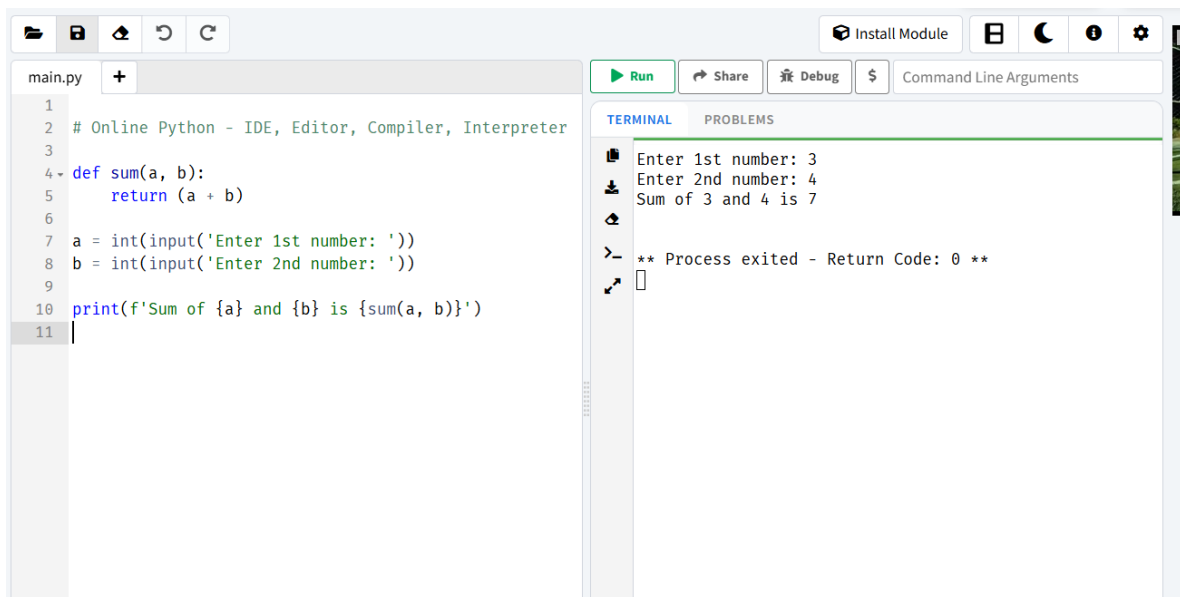
Resumen

En este capítulo hemos aprendido que:

- Programar consiste en dar instrucciones a un ordenador para resolver problemas.
- Un algoritmo es una secuencia ordenada de pasos para resolver un problema.
- Un programa es la implementación de un algoritmo en un lenguaje de programación.

- El pseudocódigo permite describir algoritmos de forma sencilla e independiente del lenguaje utilizado.
- Python es un lenguaje sencillo y muy utilizado.
- Los programas se escriben en un editor y se ejecutan para obtener resultados.
- La función `print()` permite mostrar información en pantalla.
- Los comentarios con `#` ayudan a explicar el código.
- Las tres comillas permiten escribir documentación extensa o cadenas de texto multilínea.
- Los errores forman parte del aprendizaje y ayudan a mejorar nuestros programas.

En el próximo capítulo aprenderemos a almacenar información utilizando variables.



The screenshot shows an online Python IDE interface. On the left, a code editor displays a Python script in `main.py`. The script defines a `sum` function and uses `input` and `print` to interact with the user. On the right, a terminal window shows the output of the program after execution, including user input and the calculated sum. The interface includes standard IDE controls like run, share, and debug buttons at the top.

```
1
2 # Online Python - IDE, Editor, Compiler, Interpreter
3
4 def sum(a, b):
5     return (a + b)
6
7 a = int(input('Enter 1st number: '))
8 b = int(input('Enter 2nd number: '))
9
10 print(f'Sum of {a} and {b} is {sum(a, b)}')
11
```

TERMINAL

```
Enter 1st number: 3
Enter 2nd number: 4
Sum of 3 and 4 is 7

> ** Process exited - Return Code: 0 **
```