

CAPÍTULO 5. ALGORITMOS SECUENCIALES Y DIAGRAMAS DE FLUJO

1. Introducción

Hasta ahora hemos aprendido a utilizar instrucciones básicas de Python:

- Mostrar información con `print()`.
- Almacenar datos en variables.
- Solicitar información mediante `input()`.
- Realizar operaciones matemáticas.

Sin embargo, escribir código no consiste únicamente en conocer instrucciones.

Antes de programar debemos aprender a resolver problemas.

Por ello, en este capítulo estudiaremos:

- Los algoritmos secuenciales.
 - El pseudocódigo.
 - Los diagramas de flujo.
 - La metodología de resolución de problemas.
-

2. ¿Qué es un algoritmo secuencial?

Un algoritmo secuencial es aquel cuyas instrucciones se ejecutan una detrás de otra en el mismo orden en que han sido escritas.

Por ejemplo:

Paso 1
Paso 2
Paso 3
Paso 4

Cada paso se ejecuta exactamente una vez.

Los algoritmos que hemos realizado hasta ahora son secuenciales.

Por ejemplo:

```
nombre = input("Nombre:")  
print(nombre)
```

Primero se solicita el nombre y después se muestra.

3. Metodología para resolver problemas

A partir de este momento seguiremos siempre los mismos pasos.

Paso 1. Comprender el problema

Debemos identificar:

- Qué datos conocemos.
- Qué datos necesitamos calcular.
- Qué resultado debemos mostrar.

Paso 2. Diseñar el algoritmo

Escribir los pasos necesarios para resolver el problema.

Paso 3. Escribir el pseudocódigo

Representar el algoritmo de forma estructurada.

Paso 4. Dibujar el diagrama de flujo

Representar gráficamente la solución.

Paso 5. Programar en Python

Traducir la solución al lenguaje Python.

4. El pseudocódigo

El pseudocódigo es una representación intermedia entre el lenguaje natural

y el lenguaje de programación.

Permite describir algoritmos sin preocuparnos todavía por la sintaxis de Python.

Ejemplo:

Leer nombre
Escribir nombre

Otro ejemplo:

Leer base
Leer altura

$\text{area} \leftarrow \text{base} \times \text{altura}$

Escribir area

5. Diagramas de flujo

Los diagramas de flujo representan gráficamente un algoritmo.

Utilizan símbolos estandarizados.

Inicio y fin

Se representan mediante una elipse.

Inicio

Fin

Entrada y salida

Se representan mediante un paralelogramo.

⟨ Leer edad ⟩

⟨ Mostrar edad ⟩

Proceso

Se representa mediante un rectángulo.

$$\text{area} = b \times h$$

Flujo

La dirección de ejecución se representa mediante flechas.

```
Inicio
  ↓
Leer dato
  ↓
Proceso
  ↓
Mostrar resultado
  ↓
Fin
```

6. Primer ejemplo completo

Problema

Solicitar un nombre y mostrarlo posteriormente.

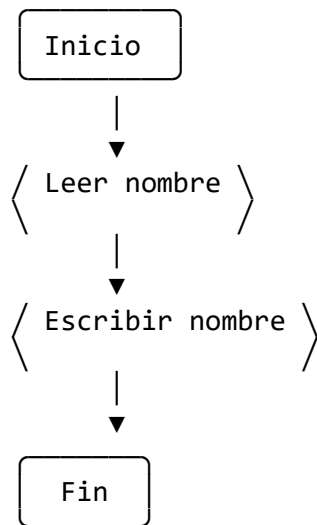
Algoritmo

1. Pedir el nombre.
 2. Guardarlo.
 3. Mostrarlo.
-

Seudocódigo

Leer nombre
Escribir nombre

Diagrama de flujo



Python

```
nombre = input("Nombre:")  
print(nombre)
```

7. Segundo ejemplo completo

Problema

Calcular el área de un rectángulo.

Algoritmo

1. Leer la base.
2. Leer la altura.
3. Calcular el área.

4. Mostrar el resultado.

Seudocódigo

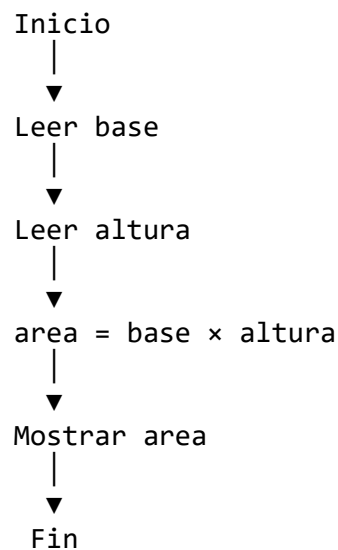
Leer base

Leer altura

area $\leftarrow$ base $\times$ altura

Escribir area

Diagrama de flujo



Python

```
base = float(input("Base:"))
```

```
altura = float(input("Altura:"))
```

```
area = base * altura
```

```
print(area)
```

8. Trazado de algoritmos

Una forma de comprobar si un algoritmo funciona consiste en seguir paso a paso la evolución de las variables.

Ejemplo:

```
a = 5
b = 3
```

```
suma = a + b
```

Tabla de seguimiento:

Instrucción	a	b	suma
a = 5	5	-	-
b = 3	5	3	-
suma = a + b	5	3	8

Esta técnica ayuda a detectar errores.

9. Ejemplos resueltos

Ejemplo 1. Perímetro de un cuadrado

Algoritmo

1. Leer lado.
2. Calcular perímetro.
3. Mostrar resultado.

Seudocódigo

Leer lado

```
perimetro ← 4 × lado
```

Escribir perimetro

Python

```
lado = float(input("Lado:"))
```

```
perimetro = 4 * lado
```

```
print(perimetro)
```

Ejemplo 2. Media de tres notas

Algoritmo

1. Leer tres notas.
2. Calcular la media.
3. Mostrar resultado.

Seudocódigo

Leer nota1
Leer nota2
Leer nota3

$media \leftarrow (nota1 + nota2 + nota3) / 3$

Escribir media

Python

```
nota1 = float(input("Nota 1:"))  
nota2 = float(input("Nota 2:"))  
nota3 = float(input("Nota 3:"))  
media = (nota1 + nota2 + nota3) / 3  
print(media)
```

10. Errores frecuentes

Error 1. Empezar a programar sin comprender el problema

Muchos errores aparecen porque el alumno comienza a escribir código demasiado pronto.

Antes de programar:

1. Comprende el problema.
2. Diseña el algoritmo.
3. Escribe el pseudocódigo.

Error 2. Saltarse pasos en el algoritmo

Un algoritmo debe incluir todas las acciones necesarias.

Error 3. No identificar correctamente las variables

Antes de programar conviene preguntarse:

- ¿Qué datos necesito?
- ¿Qué variables voy a utilizar?

11. Actividades propuestas

Actividades básicas

1. Escribe el algoritmo para calcular el doble de un número.
2. Escribe el algoritmo para calcular el perímetro de un cuadrado.
3. Escribe el algoritmo para calcular el área de un rectángulo.

Actividades de pseudocódigo

4. Escribe el pseudocódigo para calcular la media de dos notas.
5. Escribe el pseudocódigo para calcular el área de un triángulo.
6. Escribe el pseudocódigo para calcular el volumen de un cubo.

Actividades de diagramas de flujo

7. Dibuja el diagrama de flujo para calcular el doble de un número.
8. Dibuja el diagrama de flujo para calcular el área de un rectángulo.
9. Dibuja el diagrama de flujo para calcular la media de tres notas.

Actividades de programación

10. Implementa en Python los ejercicios anteriores.
11. Calcula el área de un círculo.

12. Calcula la velocidad de un vehículo.
13. Calcula el consumo medio de combustible de un coche.

Reto final

Diseña un programa que calcule el Índice de Masa Corporal (IMC).

Antes de programar:

1. Analiza el problema.
2. Diseña el algoritmo.
3. Escribe el pseudocódigo.
4. Dibuja el diagrama de flujo.
5. Implementa la solución en Python.

Dato:

$IMC = \text{peso} / \text{altura}^2$

Resumen

En este capítulo hemos aprendido:

- Qué es un algoritmo secuencial.
- Cómo resolver problemas de forma sistemática.
- Qué es el pseudocódigo.
- Cómo construir diagramas de flujo.
- Cómo realizar el seguimiento de un algoritmo.
- Cómo pasar de una idea a un programa en Python.

En el próximo capítulo aprenderemos a tomar decisiones mediante estructuras condicionales.