

CAPÍTULO 2. VARIABLES Y TIPOS DE DATOS

1. ¿Por qué necesitamos variables?

En el capítulo anterior aprendimos a mostrar información en pantalla mediante la función `print()`.

Por ejemplo:

```
print("Ana")
```

Este programa funciona correctamente, pero siempre mostrará el mismo resultado.

¿Qué ocurriría si quisiéramos almacenar un nombre, una edad o una nota para utilizarla varias veces dentro del programa?

Necesitamos una forma de guardar información.

Para ello utilizaremos las **variables**.

Las variables son uno de los elementos más importantes de cualquier lenguaje de programación.

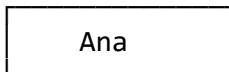
2. ¿Qué es una variable?

Una variable es un espacio de memoria donde un programa puede almacenar información.

Podemos imaginar una variable como una caja etiquetada.

Por ejemplo:

Variable: nombre



La etiqueta identifica la variable y el contenido representa el valor almacenado.

También podemos representarlo así:

nombre → "Ana"

Otro ejemplo:

edad → 17

nota → 8.5

Las variables permiten almacenar información para utilizarla posteriormente.

3. Crear una variable

Para crear una variable utilizamos el operador de asignación =.

Ejemplo:

```
nombre = "Ana"
```

En esta instrucción:

- nombre es el nombre de la variable.
- "Ana" es el valor almacenado.
- = indica que el valor debe guardarse en la variable.

Es importante comprender que el símbolo = en programación no significa igualdad matemática.

En Python significa:

Guarda el valor situado a la derecha dentro de la variable situada a la izquierda.

Otro ejemplo:

```
edad = 17
```

La variable edad almacena el número 17.

4. Mostrar el contenido de una variable

Podemos visualizar el contenido de una variable mediante la función print().

```
nombre = "Ana"
```

```
print(nombre)
```

Resultado:

Ana

Observa que ahora no utilizamos comillas dentro de `print()`.

Si escribimos:

```
print("nombre")
```

Resultado:

nombre

porque Python interpreta que queremos mostrar el texto literal “nombre”.

5. Del algoritmo al programa

A partir de este capítulo seguiremos siempre el mismo método para resolver problemas.

Problema

Guardar un nombre y mostrarlo en pantalla.

Algoritmo

1. Crear una variable llamada nombre.
 2. Guardar un nombre en ella.
 3. Mostrar el contenido de la variable.
-

Seudocódigo

```
nombre ← "Ana"  
Escribir nombre
```

Programa en Python

```
nombre = "Ana"  
  
print(nombre)
```

6. Modificar una variable

Las variables reciben este nombre porque su contenido puede variar.

```
edad = 16  
  
print(edad)  
  
edad = 17  
  
print(edad)
```

Resultado:

```
16  
17
```

Cuando asignamos un nuevo valor, el valor anterior desaparece.

7. Seguimiento paso a paso de una variable

Cuando un programa se ejecuta, las variables van cambiando de valor.

Observa el siguiente ejemplo:

```
edad = 16  
edad = 17  
edad = 18
```

La evolución de la variable es la siguiente:

Instrucción	Valor de edad
edad = 16	16
edad = 17	17
edad = 18	18

Al finalizar el programa, la variable contiene el valor 18.

Esta técnica de seguimiento recibe el nombre de **traza** o **seguimiento de ejecución**.

8. Operaciones con variables

Las variables no solo sirven para almacenar información.

También pueden utilizarse para realizar cálculos.

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

```
suma = a + b
```

```
print(suma)
```

Resultado:

8

Otro ejemplo:

```
precio = 25  
cantidad = 4
```

```
total = precio * cantidad
```

```
print(total)
```

Resultado:

100

9. Expresiones

Una expresión es una combinación de valores, variables y operadores que produce un resultado.

Por ejemplo:

```
7 + 2
```

es una expresión.

También:

```
precio * cantidad
```

o

```
nota1 + nota2 + nota3
```

Python evalúa la expresión y almacena el resultado.

Ejemplo:

```
resultado = 7 + 2
```

Primero calcula:

```
7 + 2 = 9
```

y después almacena:

```
resultado = 9
```

10. Tipos básicos de datos

La información almacenada en una variable puede ser de distintos tipos.

Cada uno de ellos recibe el nombre de tipo de dato.

10.1 Cadenas de texto (str)

Almacenan texto.

```
nombre = "Ana"  
ciudad = "Palencia"
```

Las cadenas de texto siempre se escriben entre comillas.

10.2 Números enteros (int)

Almacenan números sin decimales.

```
edad = 17  
alumnos = 30
```

10.3 Números reales (float)

Almacenan números con decimales.

```
altura = 1.72  
nota = 8.5
```

Observa que Python utiliza el punto para separar la parte decimal.

10.4 Valores lógicos (bool)

Solo pueden tomar dos valores:

```
aprobado = True
```

o

```
aprobado = False
```

Más adelante utilizaremos estos valores para tomar decisiones.

11. La función type()

La función `type()` permite conocer el tipo de una variable.

Ejemplo:

```
nombre = "Ana"
```

```
print(type(nombre))
```

Resultado:

```
<class 'str'>
```

Otros ejemplos:

```
edad = 17
```

```
print(type(edad))
```

```
nota = 8.5
```

```
print(type(nota))
```

```
aprobado = True
```

```
print(type(aprobado))
```

12. Normas para nombrar variables

Al crear variables debemos seguir ciertas reglas.

No pueden comenzar por un número

Incorrecto:

```
2nombre = "Ana"
```

Correcto:

```
nombre2 = "Ana"
```

No pueden contener espacios

Incorrecto:

```
nombre alumno = "Ana"
```

Correcto:

```
nombre_alumno = "Ana"
```

No pueden utilizar palabras reservadas

Incorrecto:

```
class = "A"
```

Correcto:

```
curso = "A"
```

Python distingue mayúsculas y minúsculas

```
nombre = "Ana"  
Nombre = "Luis"
```

Son variables distintas.

13. La convención snake_case

Los programadores utilizan ciertas convenciones para nombrar variables.

La más utilizada en Python se llama **snake_case**.

Consiste en:

- Escribir todas las palabras en minúsculas.
- Separarlas mediante guiones bajos.

Ejemplos:

```
nombre_alumno = "Ana"
nota_media = 8.5
numero_intentos = 3
fecha_nacimiento = "12/05/2008"
```

14. Buenas prácticas

Evita nombres poco descriptivos.

Poco recomendable:

```
a = "Ana"
b = 17
c = 8.5
```

Mucho mejor:

```
nombre_alumno = "Ana"
edad_alumno = 17
nota_media = 8.5
```

Un buen nombre permite comprender rápidamente qué almacena cada variable.

15. Ejemplos resueltos

Ejemplo 1. Guardar un nombre

```
nombre = "Ana"

print(nombre)
```

Ejemplo 2. Guardar varios datos

```
nombre = "Ana"
edad = 17
curso = "2º Bachillerato"

print(nombre)
print(edad)
print(curso)
```

Ejemplo 3. Realizar una suma

```
a = 5
b = 3

suma = a + b

print(suma)
```

Ejemplo 4. Calcular un área

Algoritmo

1. Guardar la base.
2. Guardar la altura.
3. Calcular el área.
4. Mostrar el resultado.

Seudocódigo

```
base ← 5
altura ← 4
area ← base × altura
Escribir area
```

Python

```
base = 5
altura = 4

area = base * altura

print(area)
```

16. Errores frecuentes

Error 1. Utilizar una variable que no existe

```
print(nombre)
```

Resultado:

NameError

Error 2. Olvidar las comillas

Incorrecto:

```
nombre = Ana
```

Correcto:

```
nombre = "Ana"
```

Error 3. Escribir espacios en el nombre

Incorrecto:

```
nombre alumno = "Ana"
```

Correcto:

```
nombre_alumno = "Ana"
```

Error 4. Confundir mayúsculas y minúsculas

```
nombre = "Ana"
```

```
print(Nombre)
```

Resultado:

NameError

Actividades propuestas

Actividades básicas

1. Crea una variable llamada nombre y almacena tu nombre.
 2. Muestra el contenido de la variable.
 3. Crea una variable llamada edad.
 4. Muestra la edad por pantalla.
 5. Crea una variable llamada ciudad.
-

Actividades de práctica

6. Crea variables para almacenar tu nombre, edad, ciudad y curso.

7. Muestra todas las variables.
 8. Crea una variable llamada temperatura con valor 25.
 9. Modifica la temperatura para que valga 30.
 10. Comprueba el tipo de varias variables mediante `type()`.
 11. Crea una variable lógica llamada aprobado.
-

Actividades de algoritmia

12. Diseña el algoritmo para almacenar y mostrar una nota.
 13. Escribe el pseudocódigo correspondiente.
 14. Implementa el programa en Python.
 15. Diseña el algoritmo para calcular el perímetro de un cuadrado de lado 8.
-

Actividades avanzadas

16. Crea variables para almacenar tres notas y calcula su suma.
 17. Calcula el precio total de cinco productos iguales.
 18. Calcula el área de un rectángulo utilizando variables.
 19. Calcula el perímetro de un rectángulo utilizando variables.
-

Reto final

Diseña un programa que almacene en variables:

- Nombre.
- Edad.
- Ciudad.
- Curso.
- Altura.
- Nota media.

Antes de programar:

1. Diseña el algoritmo.
2. Escribe el pseudocódigo.

Finalmente, implementa el programa en Python y muestra toda la información almacenada.

Resumen

En este capítulo hemos aprendido que:

- Las variables permiten almacenar información.
- El operador `=` sirve para asignar valores.
- Las variables pueden cambiar durante la ejecución.
- Es posible realizar operaciones con variables.
- Existen diferentes tipos de datos: `str`, `int`, `float` y `bool`.
- La función `type()` permite conocer el tipo de una variable.
- Python distingue mayúsculas y minúsculas.
- La convención `snake_case` facilita la lectura del código.
- Los buenos nombres de variables hacen que los programas sean más comprensibles.

En el próximo capítulo aprenderemos a introducir datos mediante el teclado utilizando la función `input()`.