

Ejercicios de Secuencias

Programación – DAW

Ricardo Pérez López
IES Doñana

Curso 2025/2026

1. Dadas las siguientes variables con valores similares a los dados:

```
nombre = "Pedro"  
edad = 35
```

imprimir una cadena como la siguiente:

Me llamo Pedro y tengo 35 años

2. Dadas las siguientes variables con valores similares a los dados:

```
a = 12  
b = 8
```

escribir una función llamada `imprimir_expresión` que devuelva una cadena como la siguiente:

12 + 8 = 20

3. Dada la siguiente variable con un valor similar al dado:

```
lenguaje = "Python"
```

imprimir una cadena como la siguiente:

Me gusta aprender PYTHON

4. Dada la siguiente variable con un valor similar al dado:

```
precio = 19.98765
```

imprimir una cadena como la siguiente:

Precio: 19.99 €

5. Dadas las siguientes variables con valores similares a los dados:

```
producto = "Manzana"  
precio = 0.5
```

imprimir en formato de tabla una cadena como la siguiente, con ancho 10 para el nombre y 6 para el precio, alineando el precio a la derecha:

Manzana 0.50

6. Dadas las siguientes variables con valores similares a los dados:

```
aciertos = 45  
total = 50
```

imprimir una cadena como la siguiente:

Has acertado 90.00%

7. Imprimir el contenido de la variable `nombre` ocupando 10 caracteres y alineado a la derecha.

8. Imprimir la siguiente tabla sencilla:

Producto	Precio
Manzana	0.50
Plátano	0.75

9. A partir de unos datos como los siguientes:

```
datos = [ ("Tomates", 5, 99.00),  
         ("Manzanas", 3, 1.40),  
         ("Cola-cao 10 kilogramos", 1, 10.00),  
         ("Impresora HP", 2, 69.00)]
```

escribir una función llamada `imprimir_factura` que imprima una salida como la siguiente:

Denominación	Cantidad	Precio	Importe

Tomates	5	99.00 €	495.00 €
Manzanas	3	1.40 €	4.20 €
Cola-cao 10 kil	1	10.00 €	10.00 €
Impresora HP	2	69.00 €	138.00 €

TOTAL: 647.20 €			
=====			

10. Dada una lista de calificaciones como la siguiente:

```
notas = [7, 8.5, 6, 9]
```

escribir una función llamada `imprimir_notas` que imprima cada nota numerada empezando por 1.

11. Dada la siguiente variable con valor similar al dado:

```
numero = 7
```

imprimir una cadena como la siguiente:

El número 7 es impar

12. Dadas las variables siguientes con valores similares a los dados:

```
a, b = 5, 8
```

imprimir una cadena como la siguiente:

$5 * 8 = 40$, y $5 + 8 = 13$

usando una sola *f-string*.

13. Dada la siguiente variable, con un valor similar al dado:

```
frutas = ["manzana", "pera", "plátano"]
```

imprimir una lista numerada como la siguiente, usando una *f-string* dentro de un bucle:

1. manzana
2. pera
3. plátano

14. Escribir una función llamada `imprimir_tabla_multiplicar` que imprima la tabla de multiplicar de un número en formato alineado con *f-strings* y ancho 2 para números:

```
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
...
7 x 10 = 70
```

15. Dada una lista de saldos con valores similares a los dados:

```
saldos = [1200, -350.5, 78, -15.75]
```

imprimir cada saldo con decimales y € al final, alineado a la derecha en columna de ancho 10.

16. Escribir una *f-string* que muestre:

La raíz cuadrada de 16 es 4.00

usando la función `math.sqrt` dentro de una *f-string*.

17. Dada una lista con valores similares a los dados:

```
notas = [7, 8.5, 6, 9]
```

mostrar la media de la lista en una sola línea, redondeada a 2 decimales.

18. Dado un texto como el siguiente:

```
texto = "Hoy es 18 de enero de 2026"
```

escribir una expresión regular que encuentre **todos los números** en la cadena.

19. Dado un texto como el siguiente:

```
frase = "Me gustan los gatos y los perros, pero no los ratones"
```

encontrar todas las apariciones de “gatos” o “perros” en la cadena.

20. Dada una lista de direcciones de e-mail similar a la siguiente:

```
correos = [
    "usuario123@gmail.com",
    "usuario@dominio.es",
    "mal.email@dominio"
]
```

comprobar cuáles cumplen el patrón básico `nombre@dominio.extension` (con `.com` o `.es`).

21. Dada una lista de fechas similar a la siguiente:

```
fechas = ["01/12/2023, 15/01/2026, 3/7/2025, 31/2/2026"]
```

comprobar cuáles cumplen el patrón dd/mm/yyyy.

22. Dada una cadena similar a la siguiente:

```
frase = "Un código postal es 11001, el otro es 543, y hay otro que es el 28013."
```

encontrar todos los códigos postales válidos.

23. Dada una lista de teléfonos similar a la siguiente:

```
telefonos = ["+34 600-123-456", "600123456", "123-456-789"]
```

verificar cuáles cumplen el patrón de **teléfono español válido**.

24. Dado un texto como el siguiente:

```
texto = "Hoy Lunes es un buen Día para Aprender Python"
```

encontrar todas las palabras que **empiecen con letra mayúscula**.

25. Dado un texto como el siguiente:

```
texto = "Educación, programación, función, gato, acción"
```

encontrar todas las palabras que **terminen en “ción”**.

26. Dado un texto como el siguiente:

```
texto = "aeropuerto, cuevas, baúl, rio"
```

encontrar todos los lugares donde hay **dos o más vocales consecutivas**.

27. Dado un texto como el siguiente:

```
texto = "Galleta, llama, perro, puerta, silla"
```

encontrar todas las palabras que contengan **letras repetidas consecutivas** (por ejemplo, **ll**, **rr**, **ll**).

28. Dado un texto como el siguiente:

```
texto = "Hola123 mundo, prueba_1, test"
```

encontrar todas las palabras que **contengan solo letras**.

29. Dado un texto como el siguiente:

```
texto = "uno dos tres cuatro cinco seis siete"
```

encontrar todas las palabras de **exactamente 3 letras**.

30. Dado un texto como el siguiente:

```
texto = "Hay 3 gatos y 12 perros en la casa"
```

encontrar **solo el primer número** en la cadena.

31. Dado un texto como el siguiente:

```
texto = "Python es un lenguaje, tipografía y pyme"
```

encontrar todas las palabras que **contengan py** (mayúsculas o minúsculas).

32. Dado un texto como el siguiente:

```
texto = "hola holanda holá holi"
```

comprobar si la palabra **hola** aparece como **palabra completa**, no como parte de otra palabra.