Iterar en el lenguaje **Python** es el proceso de ejecutar un bloque de código repetidamente, permitiendo recorrer estructuras de datos (como listas, diccionarios o tuplas) o realizar tareas cíclicas de manera eficiente.

A continuación, presento el concepto y los ejemplos bajo un estándar académico:

Concepto de Iteración

En Python, la iteración se basa principalmente en el protocolo del **iterable**. Un objeto es iterable si puede devolver sus elementos uno a la vez, permitiendo que estructuras de control como los bucles procesen cada dato de forma secuencial.

- **Bucle for:** Se utiliza para iteraciones definidas, donde el número de repeticiones se conoce de antemano o depende de la longitud de una colección.
- **Bucle while:** Se utiliza para iteraciones condicionales, ejecutándose mientras una expresión lógica sea verdadera ($True$).

Ejemplos Prácticos

1. Iteración sobre una Lista (Bucle for)

Es la forma más común de recorrer una colección de elementos.

Python
```
frutas = ["manzana", "pera", "uva"]

for fruta in frutas:
    print(f"Hoy quiero comer {fruta}")
```

2. Uso de la función range()

Ideal para ejecutar un ciclo un número específico de veces sin necesidad de una lista previa.

Python
```
# Genera números del 0 al 4
for i in range(5):
    print(f"Iteración número: {i}")
```

3. Iteración Condicional (Bucle while)

Este ciclo continúa hasta que el contador ya no cumple la condición establecida.

Python
```
contador = 1

while contador <= 3:
    print(f"Contando: {contador}")
```

```
    contador += 1  # Incremento fundamental para evitar bucles infinitos
```

4. Iteración sobre Diccionarios

Python permite iterar tanto en las llaves como en los valores de un diccionario.

```
Python
notas = {"Luis": 18, "Ana": 20, "Marta": 15}

for nombre, calificacion in notas.items():
    print(f"{nombre} obtuvo una nota de {calificacion}")
```

Aquí tienes una serie de ejercicios diseñados para practicar la lógica de bucles definidos (for) e indefinidos (while), escalando desde conceptos básicos hasta aplicaciones prácticas similares a la lógica de la calculadora que analizamos anteriormente.

Práctica de Bucles while (Control por Condición)

1. **Validación de Entrada:** Crea un programa que solicite una contraseña al usuario y no se detenga hasta que la entrada sea igual a "Python123".
2. **Calculadora Acumulativa:** Escribe un bucle que pida números al usuario y los sume. El ciclo debe terminar cuando el usuario ingrese el número 0.
3. **Control de Inventario:** Imagina que tienes 50 productos en stock. Crea un while que reste cantidades vendidas solicitadas por el usuario hasta que el stock llegue a menos de 5.
4. **Adivina el Número:** Genera un número aleatorio entre 1 y 10. Pide al usuario que intente adivinarlo; el bucle debe continuar hasta que el usuario acierte.
5. **Simulador de Cajero:** Crea un programa que solicite un PIN. El usuario tiene solo 3 intentos. Si falla los 3, el bucle termina con un mensaje de "Cuenta bloqueada".

Práctica de Bucles for e Iteración (Control por Colección)

6. **Generador de Tablas:** Utiliza un for y la función range() para imprimir la tabla de multiplicar de un número ingresado por el usuario (del 1 al 10).
7. **Filtro de Listas:** Dada una lista de calificaciones [15, 20, 12, 18, 9, 14, 10], itera sobre ella e imprime solo aquellas que sean mayores o iguales a 15 (aprobados).
8. **Contador de Vocales:** Solicita una frase al usuario e itera sobre cada carácter para contar cuántas vocales (a, e, i, o, u) contiene.
9. **Formateador de Datos (Estilo Calculadora):** Dada una lista de nombres de botones ["Limpiar", "Print", "Suma", "Igual"], utiliza un bucle para imprimir: "Configurando botón: [Nombre]".
10. **Inversor de Cadenas:** Crea un programa que tome una palabra e itere sobre ella de forma inversa para mostrarla al revés (ejemplo: "Python" -> "nohtyP").
