

Comprensión de listas

Ejercicio: Crear una lista de los números pares del 1 al 10

Este código utiliza una comprensión de listas para generar una lista de números pares del 1 al 10 y luego imprime la lista resultante.

- **Generación de la Lista:**

- Se define una lista llamada *pares* utilizando una comprensión de listas. La sintaxis de la comprensión de listas con una condición es: *[expresión for elemento in iterable if condición]*.
- En este caso, la expresión es *elemento*, lo que significa que se añaden a la lista los valores de *elemento* que cumplen con la condición especificada.

- **Expresión en la Comprensión de Listas:**

- La parte *for elemento in range(1, 11)* define un bucle que itera sobre un rango de números del 1 al 10 (inclusive).
- *range(1, 11)* genera los números 1, 2, 3, ..., hasta 10.
- La condición *if elemento % 2 == 0* filtra los números para incluir solo aquellos que son pares.

- **Primera Iteración:**

- En la primera iteración, *elemento* toma el valor 1.
- La condición *1 % 2 == 0* se evalúa como falsa (porque 1 no es divisible entre 2 sin residuo).
- Por lo tanto, 1 no se añade a la lista *pares*.

- **Siguientes Iteraciones:**

- El bucle continúa para los valores de *elemento* desde 2 hasta 10:
 - * Para *elemento = 2*, *2 % 2 == 0* se evalúa como verdadera (porque 2 es divisible entre 2 sin residuo).
 - * El valor 2 se añade a la lista *pares*.

- * Para $\text{elemento} = 3$, $3 \% 2 == 0$ se evalúa como falsa (porque 3 no es divisible entre 2 sin residuo).
- * Por lo tanto, 3 no se añade a la lista.
- * Este proceso continúa hasta que elemento alcanza 10, donde $10 \% 2 == 0$ se evalúa como verdadera y el valor 10 se añade a la lista.

- **Lista Final:**

- La lista *pares* contiene ahora los valores: $[2, 4, 6, 8, 10]$.
- Cada elemento de la lista es un número par en el rango del 1 al 10.

- **Impresión de la Lista:**

- Finalmente, la lista *pares* se imprime utilizando `print(pares)`.
- La salida en consola será: $[2, 4, 6, 8, 10]$.

```
pares = [elemento for elemento in range(1, 11) if elemento % 2 == 0]
print(pares)
```

$[2, 4, 6, 8, 10]$

Ejercicio: Crear una lista de las longitudes de las palabras en una lista

Este código utiliza una comprensión de listas para generar una lista que contiene las longitudes de las palabras en una lista dada y luego imprime la lista resultante.

- **Generación de la Lista:**

- Se define una lista llamada *longitudes* utilizando una comprensión de listas. La sintaxis de la comprensión de listas es: $[\text{expresión for elemento in iterable}]$.
- En este caso, la expresión es $\text{len}(\text{palabra})$, que calcula la longitud de cada palabra en la lista original.

- **Expresión en la Comprensión de Listas:**

- La parte $\text{for palabra in palabras}$ define un bucle que itera sobre cada elemento en la lista *palabras*.
- La lista *palabras* contiene las cadenas: $["manzana", "banana", "cereza"]$.

- **Iteraciones de la Comprensión de Listas:**

- **Primera Iteración:**

- * En la primera iteración, *palabra* toma el valor "manzana".
- * Se calcula la longitud de "manzana" usando $\text{len}(\text{"manzana"})$, que es 7.
- * El valor 7 se añade a la lista *longitudes*.

- **Segunda Iteración:**
 - * En la segunda iteración, *palabra* toma el valor “banana”.
 - * Se calcula la longitud de “banana” usando *len(“banana”)*, que es 6.
 - * El valor 6 se añade a la lista *longitudes*.
- **Tercera Iteración:**
 - * En la tercera iteración, *palabra* toma el valor “cereza”.
 - * Se calcula la longitud de “cereza” usando *len(“cereza”)*, que es 6.
 - * El valor 6 se añade a la lista *longitudes*.
- **Lista Final:**
 - La lista *longitudes* contiene los valores: *[7, 6, 6]*.
 - Cada elemento de la lista representa la longitud de la correspondiente palabra en la lista *palabras*.
- **Impresión de la Lista:**
 - Finalmente, la lista *longitudes* se imprime utilizando *print(longitudes)*.
 - La salida en consola será: *[7, 6, 6]*.

```
palabras = ["manzana", "banana", "cereza"]
longitudes = [len(palabra) for palabra in palabras]
print(longitudes)
```

[7, 6, 6]