Conceptos avanzados de listas

List comprehensions

Hemos observado en la resolución de problemas que generalmente utilizamos “el mismo código” (similar) para resolver diversas cuestiones que se repiten continuamente.

Particularmente en los ejercicios de repetición con listas, hemos utilizado el siguiente patrón que se repite en diferentes circunstancias:

```
nueva_lista = [] #Creo lista vacía

#recorro todos los elementos de la lista original
for valor in lista_original:
    #si el valor cumple un criterio, lo agrego a otra lista
    if valor % 2 == 0:
        nueva_lista.append(valor)
# En este punto, nueva_lista tiene los valores pares
```

List comprehensions

El objetivo del concepto de List Comprehensions en el lenguaje Python es facilitarnos la tarea de transformación de listas.

Transformar una lista significa que podemos:

- Aplicar una transformación a cada elemento. Ej: Elevar al cuadrado cada elemento de la lista.
- Filtrar los elementos de una lista. Ej: Filtrar solamente los valores positivos.
- Combinación de las anteriores: Ej: Multiplicar por 10 los valores impares de una lista.

List comprehensions

Todos los operaciones que se resuelven usando “List Comprehensions” pueden ser implementadas con un ciclo “for”. La inversa NO es válida.

```
nueva_lista = []
for valor in lista_original:
    if valor % 2 == 0:
        nueva_lista.append(valor)
```

```
newlist = [expression for item in iterable if condition == True]
nueva_lista =[valor for valor in lista_original if valor % 2 == 0]
```

Ejemplo 1

Aplicar una **transformación** a cada elemento. Ej: Elevar al cuadrado cada elemento de la lista.

```
lista_original = [1,3,5,6,0]

lista_nueva = []

for elem in lista_original:

    lista_nueva.append(elem**2)

# podemos hacer lo mismo con List Comprehension:

lista_nueva = [elem**2 for elem in lista_original]
```

Ejemplo 2

Filtrar los elementos de una lista. Ej: Filtrar solamente los valores positivos.

```
lista_original = [1, -3, 5, -6, 0]

lista_nueva = []

for elem in lista_original:
    if elem >= 0:
        lista_nueva.append(elem)


lista_nueva = [elem for elem in lista_original if elem>=0]
```

Ejemplo 3

Combinación: Ej: Multiplicar por 10 los valores pares de una lista.

```
lista_original = [1, 3, 5, 6, 0]

lista_nueva = []

for elem in lista_original:
    if elem % 2 != 0:
        lista_nueva.append(elem * 10)


lista_nueva= [elem * 10 for elem in lista_original if elem % 2 <> 0]
```

Ejemplo 4

Mostrar los cuadrados de los números del 1 al 5:

```
lista_cuadrados = []
for i in range(1, 6):
        lista_cuadrados.append(i**2)


lista_cuadrados=[numero**2 for numero in range(1,6)]
```

Ejemplo 5

Crear una lista de palabras minúsculas a partir de una lista de palabras:

```
palabras = ["HOLA", "MUNDO", "Python", "es", "genial"]

palabras_minusculas = []

for palabra in palabras:

        palabras_minusculas.append(palabra.lower())


palabras_minusculas = [palabra.lower() for palabra in palabras]
```

```
numbers = []
for i in range(100):
    numbers.append(i)
```

```
numbers = [i for i in range(100)]
```