Práctica Enumerador 1

Imprime en pantalla frases como la siguiente:

`'{nombre} se encuentra en el índice {indice}'`

Donde ***nombre*** debe ser cada uno de los nombres de la lista a continuación, y el ***índice***, obtenido mediante `enumerate()`.

```
lista_nombres = ["Marcos", "Laura", "Mónica", "Javier", "Celina", "Marta", "Darío", "Emiliano", "Melisa"]
```

```
lista_nombres = ["Marcos", "Laura", "Mónica", "Javier", "Celina", "Marta", "Darío", "Emiliano", "Melisa"]

print(f'{nombre} se encuentra en el índice {indice}')
```

Puedes modificar la línea print() otorgada como ejemplo, pero *las frases entregadas deberán ser iguales.*

Tip: utiliza loops!

Práctica Enumerador 2

Crea una **lista** formada por las **tuplas (indice, elemento)**, formadas a partir de obtener mediante **enumerate()** los índices de cada caracter del string "**Python**".

Llama a la lista obtenida con el nombre de variable `lista_indices` .

Práctica Enumerador 3

Imprime en pantalla únicamente los **índices** de aquellos nombres de la lista a continuación, que **empiecen con M**:

```
lista_nombres = ["Marcos", "Laura", "Mónica", "Javier", "Celina", "Marta", "Darío", "Emiliano", "Melisa"]
```

Puedes resolverlo de diferentes maneras, pero servirá que tengas presente todos o algunos de los siguientes elementos:

- Loops
- Condicionales if
- El método enumerate()
- Métodos de strings o indexado

Práctica Zip 1

Muestra en pantalla frases como la del siguiente ejemplo:

La capital de Alemania es Berlín

Utiliza la función zip, loops, y las siguientes listas de países y capitales para resolverlo rápida y eficientemente.

```
capitales = ["Berlín", "Tokio", "París", "Helsinki", "Ottawa", "Canberra"]
paises = ["Alemania", "Japón", "Francia", "Finlandia", "Canadá", "Australia"]
```

Práctica Zip 2

Crea un objeto zip formado a partir de listas, de un conjunto de marcas y productos que tú prefieras, dentro de la variable `mi_zip`.

Práctica Zip 3

Crea el zip con las traducciones los números del 1 al 5 en español, portugués e inglés (*en el mismo orden*), y convierte el objeto generado en una **lista** almacenada en la variable `numeros`:

1. uno / um / one
2. dos / dois / two
3. tres / três / three
4. cuatro / quatro / four
5. cinco / cinco / five

El resultado deberá seguir la estructura:

```
[('uno', 'um', 'one'), ('dos', 'dois', 'two'), ... ]
```

Práctica Min y Max 1

Obtén el valor máximo entre los valores de la siguiente lista, y almacénalo en una variable llamada `valor_maximo`:

```
lista_numeros = [44542247/2, 21310/5, 2134747*33, 44556475, 121676, 6654067, 353254, 123134, 55**12, 611**5]
```

Práctica Min y Max 2

Calcula la diferencia entre el valor máximo y el mínimo en la siguiente lista de números, y almacénalo en una variable llamada `rango`:

```
lista_numeros = [44542247, 21310, 2134747, 44556475, 121676, 6654067, 353254, 123134, 552512, 611665]
```

Práctica Min y Max 3

Utilizando max(), min() y métodos de diccionarios, obtén el mínimo ***valor*** a partir del siguiente diccionario:

```
diccionario_edades = {"Carlos":55, "María":42, "Mabel":78, "José":44, "Lucas":24, "Rocío":35, "Sebastián":19, "Catalina":2,"Darío":49}
```

Almacena dicho valor en una variable llamada `edad_minima`.

También, obtén el ***nombre*** que se ubica último en orden alfabético, y almacénalo en una variable llamada `ultimo_nombre`.