**Listas en Python**

**Alumno**

Santiago Vázquez

**Introducción**

En esta tarea de investigación se explicarán y ejemplificarán algunos métodos útiles para trabajar con las listas en el lenguaje Python, así como otras diversas funciones para iterables que pueden ser aplicadas a las listas.

**Palabras Clave**

Python, Listas, Métodos, Funciones, Útiles, Iterables

**Desarrollo**

Antes de definirlos y ejemplificarlos, es necesario repasar el porqué de la gran utilidad de estas funciones.

Debido a que las mismas son herramientas proporcionadas por el mismo lenguaje, son porciones de código intrínseco que realizan una tarea específica de forma eficiente y conveniente. Por lo que, son muy útiles para manipular y procesar grandes conjuntos de datos en objetos iterables.

**¿Qué es un objeto iterable?**

Son objetos sobre los cuales se puede iterar, esto significa que estos objetos devuelven un valor a la vez, permitiendo así ser usados en un loop *for ... in*.

A partir de ahora todas las partes de código presentadas serán parte de un mismo código, solo que se dividirá en fragmentos como ejemplos

*class Materia:*

*\_\_alumnos: int*

*\_\_nombre: str*

*def \_\_init\_\_(self, nombre: str, alumnos: int):*

*self.\_\_alumnos = alumnos*

*self.\_\_nombre = nombre*

*def get\_alumnos(self) -> int:*

*return self.\_\_alumnos*

*def get\_nombre(self) -> str:*

*return self.\_\_nombre*

*def \_\_repr\_\_(self) -> str:*

*return f"{self.\_\_nombre} ({self.\_\_alumnos} alumnos)"*

**Función *map(function, iterable, ...)* para iterables**

Esta función devuelve un iterable cuyos valores son los que devuelve la función function cuando es ejecutada por cada elemento del iterable, ejemplo:

*iterableNombres = map(lambda x: x.get\_nombre(), lista)*

*print('Resultado: ')*

*for elemento in iterableNombres:*

*print(elemento)*

*"""*

*Consola:*

*Algoritmos y resolución de problemas*

*Programación procedural*

*Programación orientada a objetos*

*"""*

**Metodo sorted**

Esta función permite ordenar una lista de objetos utilizando un atributo específico de la clase.

*materias = [Materia("Algoritmos Numéricos ", 40),*

*Materia("Estructura de Datos y Algoritmos ", 50),*

*Materia("Paradigmas de Lenguajes ", 45),*

*Materia("Programación orientada a objetos ", 70)]*

*materias\_ordenadas\_por\_alumnos = sorted(materias, key=lambda m: m.get\_alumnos())*

*for materia in materias\_ordenadas\_por\_alumnos:*

*print(materia.get\_nombre(), materia.get\_alumnos())*

*"""*

*Consola:*

*Algoritmos Numéricos (40 alumnos)*

*Paradigmas de Lenguajes (45 alumnos)*

*Estructura de Datos y Algoritmos (50 alumnos)*

*Programación orientada a objetos (70 alumnos)*

*"""*

**Método reverse**

Este método invierte el orden de elementos en la lista, ejemplo:

*lista.reverse()*

*print('Resultado: ')*

*for elemento in lista:*

*print(elemento)*

*"""*

*Consola:*

*Programación orientada a objetos (70 alumnos)*

*Estructura de Datos y Algoritmos (50 alumnos)*

*Paradigmas de Lenguajes (45 alumnos)*

*Algoritmos Numéricos (40 alumnos)*

*"""*

**Método *list.extend(iterable)***

Añade a la lista todos los valores devueltos por el iterable, ejemplo:

*lista.extend([*

*Materia('Estructura de Datos y Algoritmos', 50),*

*Materia('Paradigmas de Lenguajes', 45),*

*Materia('Algoritmos Numéricos', 40),*

*])*

*print('Resultado:')*

*for materia in lista:*

*print(materia)*

*"""*

*Consola:*

*Algoritmos y resolución de problemas (120 alumnos)*

*Programación procedural (90 alumnos)*

*Programación orientada a objetos (70 alumnos)*

*Estructura de Datos y Algoritmos (50 alumnos)*

*Paradigmas de Lenguajes (45 alumnos)*

*Algoritmos Numéricos (40 alumnos)*

*"""*

**Conclusiones**

En este informe se presentaron apenas unas pocas de las muchas funciones para iterables y métodos de funciones, es simple de ver que en Python hay cientos de formas proporcionadas por el propio lenguaje para interactuar con las listas. Por mencionar algunas otras:

append(): Agrega un elemento al final de la lista.

insert(): Inserta un elemento en una posición específica de la lista.

remove(): Elimina el primer elemento de la lista que coincide con el valor especificado.

pop(): Elimina y devuelve el elemento en una posición específica de la lista.

index(): Devuelve la posición del primer elemento de la lista que coincide con el valor especificado.

count(): Devuelve el número de veces que aparece un valor en la lista.

Además, siempre está la posibilidad de incluir librerías que añadan más funciones tales como la librería: re, string, unicodedata, etc.

**Bibliografía**

[1] [docs.python.org/datastructures](https://docs.python.org/es/3/tutorial/datastructures.html)

[2] [w3schools/python](https://www.w3schools.com/python/python_ref_list.asp)

[3] [docs.python.org/functions](https://docs.python.org/3/library/functions.html)

[4] [pythonlikeyoumeanit.com/iterables](https://www.pythonlikeyoumeanit.com/Module2_EssentialsOfPython/Iterables.html)