**Ejemplo de Grafo:**

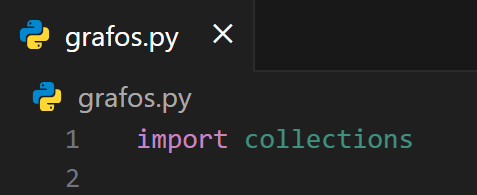
Crear un grafo simple que represente conexiones entre ciudades.

Ciudades y conexiones:

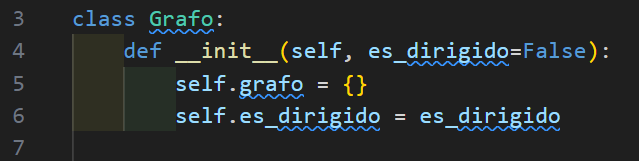
* Ciudades (Vértices): Managua, Masaya, Granada, Rivas, León
* Conexiones (Aristas):
  + Managua - Masaya
  + Managua - León
  + Masaya - Granada
  + Granada – Rivas

Vamos a usar una representación de lista de adyacencia usando un diccionario de Python. Cada clave del diccionario será un vértice, y su valor será una lista (o un conjunto, para mayor eficiencia en búsquedas) de los vértices adyacentes a él.

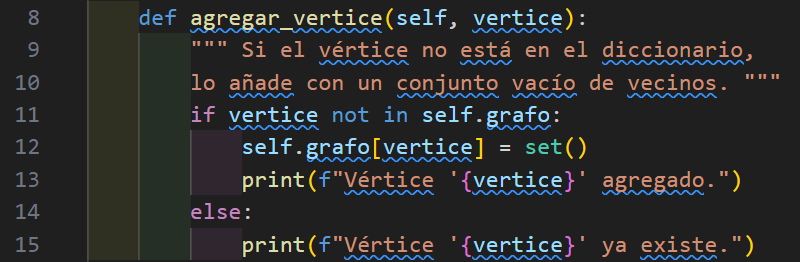
* **Paso 0: Preparación (Importar si es necesario)** Para la cola en BFS, usaremos collections.deque.



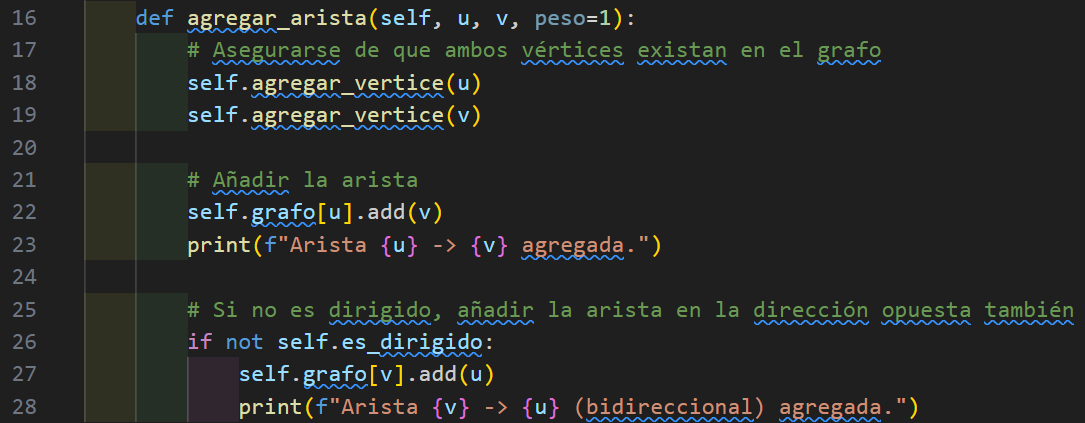
* **Paso 1: Definir la Clase Grafo y su Constructor**. Empezaremos definiendo la estructura básica de nuestra clase Grafo. El constructor inicializará un diccionario vacío que almacenará la lista de adyacencia de nuestro grafo. También añadiremos un atributo es\_dirigido para controlar el tipo de grafo.



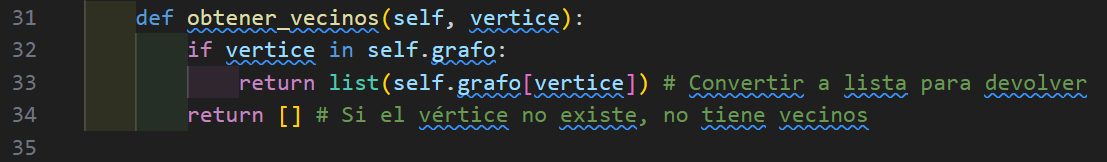
* **Paso 2: Método agregar\_vertice**. Este método se encarga de añadir un nuevo vértice a nuestro grafo. Si el vértice ya existe, simplemente no hace nada.



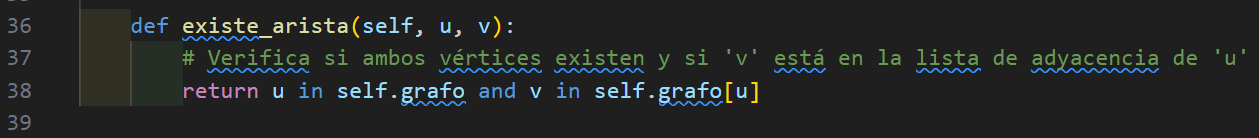
**Paso 3: Método agregar\_arista.** Este es el corazón de la construcción del grafo. Conecta dos vértices. Es crucial manejar si el grafo es dirigido o no.



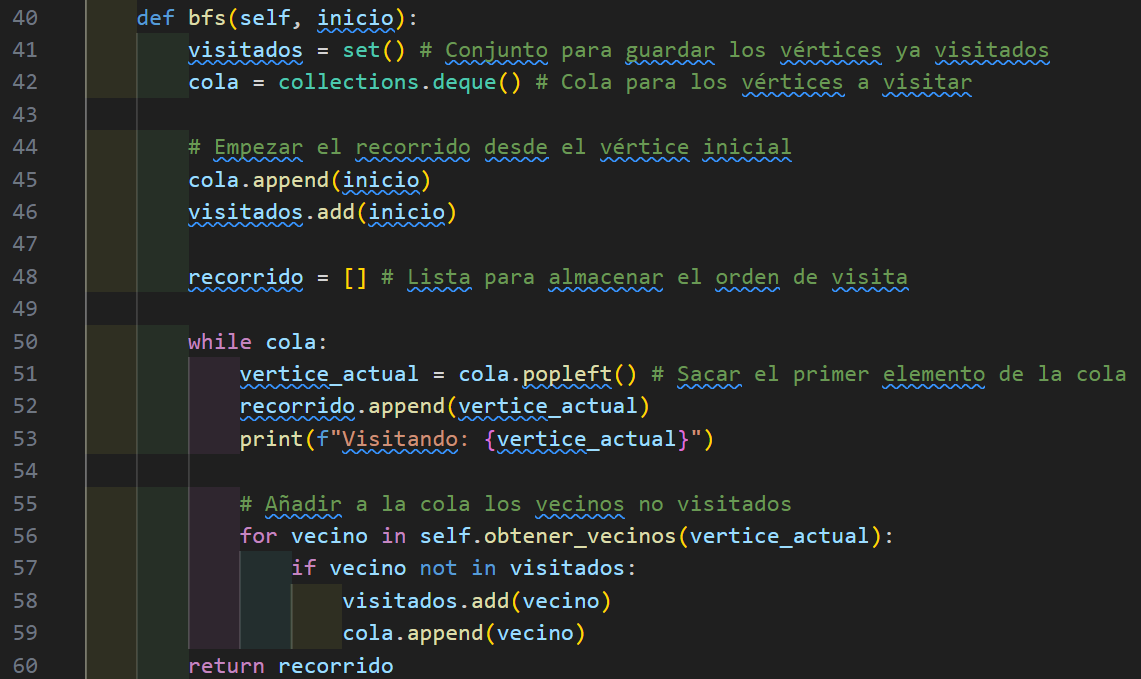
**Paso 4: Método obtener\_vecinos.** Permite consultar los vecinos de un vértice específico.



**Paso 5: Método existe\_arista**. Verifica rápidamente si hay una conexión directa entre dos vértices.

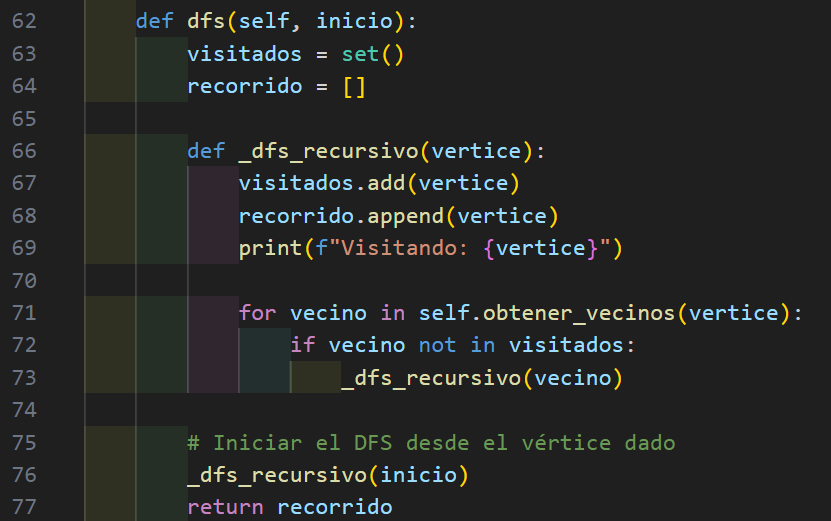


**Paso 6: Algoritmo de Búsqueda en Amplitud (BFS).** BFS explora el grafo "por niveles" o "capas" desde un vértice inicial. Utiliza una cola.

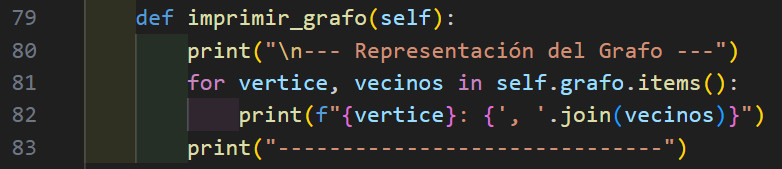


**Paso 7: Algoritmo de Búsqueda en Profundidad (DFS)**

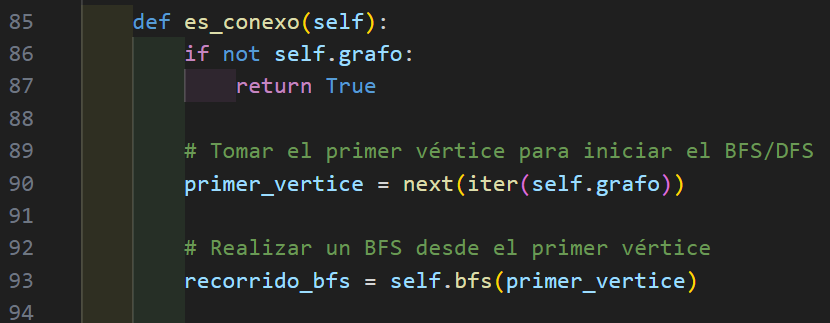
DFS explora lo más profundo posible a lo largo de cada rama antes de retroceder. Se puede implementar de forma recursiva o con una pila. Aquí usaremos la recursión que es más intuitiva.



**Paso 8. Imprimir grafo.**



**Paso 9. Función es\_conexo**

****

**Paso 10. Función encontrar\_camino**

