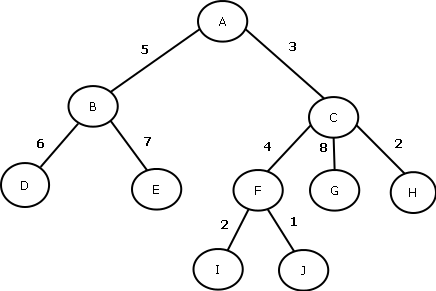
**EVALUACION PRACTICA TEORIA DE GRAFOS**

**Momento 1**

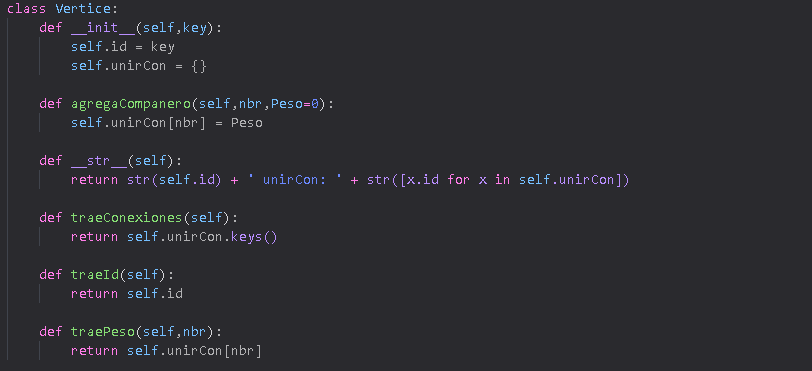
Dando alcance a lo solicitado en el Momento 1 de la evaluación práctica, utilizamos el lenguaje Python para implementar un grafo de tipo abstracto compuesto por 10 Vértices y 9 Aristas, tal y como se evidencia en la imagen a continuación:



Para implementar el grafo procedimos a crear la clase **Grafo**, la cual contiene la lista principal de los vértices y la clase **Vertice**, la cual representa cada uno de los vértices que componen el grafo. Es así que:

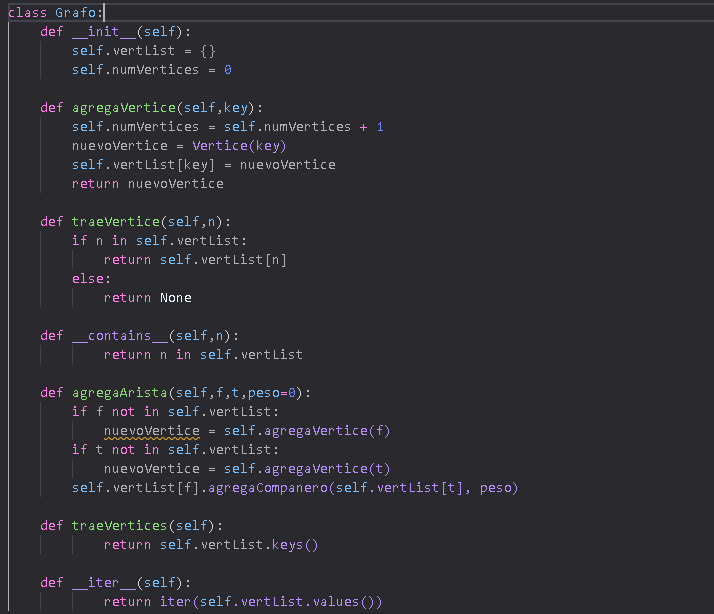
En la clase **Vertice**, cada vértice utiliza el diccionario denominado **unirCon** para así validar cada uno de los vértices a los que está conectado con sus respectivos pesos,

* El constructor inicializa el **id** (el cual puede ser un String o un Entero) y el diccionario **unirCon**.
* El método **agregaCompanero** es utilizado para agregar una conexión entre un vértice y el otro.
* El método **traeConexiones** devuelve todos los vértices en la lista de adyacencia tal como está representado en la variable de la instancia del método **unirCon**.
* El método **traePeso** retorna el peso de la Arista desde el vértice actual hasta el vértice pasado como parámetro.

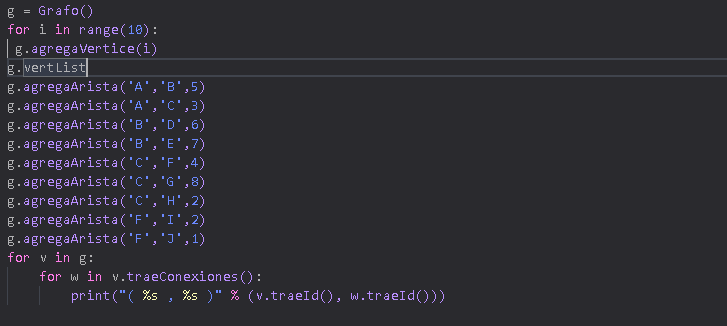


Por otro lado, la clase **Grafo** contiene un diccionario el cual mapea los nombres de las vértices a objetos,

* contiene métodos **agregaVertice** para agregar vértices al grafo y conectar un vértice a otro.
* El método **traeVertice** devuelve los nombres de todos los vértices incluidos en el grafo. Asimismo se implementó el método **\_\_iter\_\_** el cual se encarga de la iteración a través de los objetos (vértices) incluidos en un grafo.



Para representar el grafo mostrado al principio de este documento, se crea el diccionario **Vertice** compuesto por las 10 instancias de la clase Vertice. Luego se agregan cada una de las Aristas que conectan los vértices para luego correr un ciclo anidado el cual valida que cada Arista haya quedado almacenada de manera correcta.



Este es el resultado posterior a correr el código anteriormente descrito:

