Directed Graph Documentation

The graph will be read from a file in the following format:

- First line contains the number of vertices (x) and the number of edges (y)
- The following y lines contain the information of an edge (starting edge, destination and cost)

The graph is represented using *2 dictionaries* of inbound and outbound neighbours for each node. The keys of each dictionary are vertices of the graph and the values associated with them are lists of inbound/outbound neighbours.

The *costs of edges* are also stored in a dictionary which takes as a key a tuple with the following form $(startVertex,\ destinationVertex)$ and associates it with an integer (the costs of that edge).

For efficiency reasons, the graph also has 2 variables that keep track of the number of edges and vertices.

The complexities of each operation are as follows:

- O(deg(*x*) + deg(*y*)): verifying the existence of an edge and for retrieving the edge between two given vertices.
- O(1): getting the first or the next edge, inbound or outbound to a given vertex; get the endpoints, get or set the attached integer for an edge; get the total number of vertices or edges; get the in-degree or the out-degree of a given vertex.

Example: let's consider the following graph

5 7
0 1 4
1 2 7
1 5 5
2 1 4
2 3 -3
4 2 -1
5 2 2

self._nrOfEdges = 8

self._nrOfVertices = 5

self._IN = { **0:[], 1:[0,2], 2:[1,4], 3:[2], 4:[], 5:[1]** }

self._OUT = { **0:[1], 1:[2, 5], 2:[1], 3:[], 4:[2], 5:[2]** }

self._costs = { **(0, 1):4, (1, 2):7, (1, 5):5, (2, 1):4, (2, 3):-3, (4, 2):-1, (5, 2):2** }