**Documentation**

This Python code defines a Graph module that provides functionalities for creating, manipulating, and analyzing directed graphs. The module implements a Graph class with methods for adding/removing vertices and edges, accessing neighbors, retrieving costs of edges, modifying edge costs, removing vertices and edges, and creating deep copies of graphs.

**Class: Graph**

**Constructor: \_\_init\_\_(self, n=0)**

* Initializes a directed graph with **n** vertices numbered from 0 to n-1 and no edges.

**Method: add\_vertex(self, x)**

* Inserts a new vertex **x** into the graph.

**Method: add\_edge(self, x, y, cost)**

* Adds a directed edge from vertex **x** to vertex **y** with the specified **cost**.
* Returns True if the edge is added successfully, otherwise False if the edge already exists.

**Method: add\_edge\_first(self, x, y, cost)**

* Adds a directed edge from vertex **x** to vertex **y** with the specified **cost**.
* Returns True if the edge is added successfully, otherwise False if the edge already exists.

**Method: is\_edge(self, x, y)**

* Checks if there is an edge from vertex **x** to vertex **y** in the graph.
* Returns True if the edge exists, otherwise False.

**Method: parse\_nout(self, x)**

* Returns an iterable containing all outbound neighbors of vertex **x**.

**Method: parse\_nin(self, x)**

* Returns an iterable containing all inbound neighbors of vertex **x**.

**Method: parse\_vertices(self)**

* Returns an iterable containing all vertices of the graph.

**Method: get\_cost(self, x, y)**

* Returns the cost associated with the edge from vertex **x** to vertex **y**.

**Method: modify\_cost(self, x, y, c)**

* Modifies the cost of the edge from vertex **x** to vertex **y** to the specified value **c**.

**Method: remove\_vertex(self, x)**

* Removes the vertex **x** from the graph along with all its incident edges.

**Method: remove\_edge(self, x, y)**

* Removes the directed edge from vertex **x** to vertex **y**.

**Method: copy(self)**

* Returns a deep copy of the graph.

**Implementation Details**

* The Graph class utilizes dictionaries to store inbound and outbound edges of each vertex, as well as the cost of each edge.
* Methods are provided for adding/removing vertices and edges, accessing neighbors, retrieving/modifying edge costs, and creating deep copies of graphs.

This documentation provides an overview of the Graph module, its functionalities, and its methods, enabling users to effectively utilize and manipulate directed graphs through the provided interface.