Directed Graph Structure

Structure written in Python 3.6.

The DirectedGraph class has the following members:

- *vertices* – the connection that we make
  - These are represented as a list of *Vertices* (another separe class)
- *numberOfVertices* – the number of vertices that we have
- numberOfEdges -the number of edges that we have

The Vertices class has the following members:

- *edges* – a list of *edges*, the connections that there are connected to the vertex
- *inboundEdge* – a list of the in bound edges that are connected to the vertex
- *outDegree* – the out degree of the vertex
- *inDegree* – the in degree of the vertex
  - All members have getters inside the class

The Edge class has the following members:

- *cost* – the cost of the edge
- *destination* – a list of the destinations that the edge has

Graph class functions:

- readVertices – will get as an argument a file path.
  - This function will read the file and then will store the connections. The alghoritm works like this:

  1. The function will open that file and will expect: number of vertices, followed by a space then the number of edges, then followed the edges stored in the formation: point that starts the edge, point from which it end then the cost of that edge.
  2. Will read the files and will store them in the class member: ***vertices***
  3. Everytime we read an ending point, we increase the in degree of it and will add an inbound connection
  4. Then we close the file.
- GetNumberOfVertices – will return the first value of the file
- checkIfEdge – will receive 2 parameters: positionX and positionY. There will represent the starting point, ending point of an edge.
  - The program will check if there exists an edge connected from positionX to positionY

  1. Will iterate trough all connections from positionX and will check if there is a vertex equal to positionY

        2. If it is will return 1. Otherwise will return -1

- getInDegree, getOutDegree – will receive a number, which represents the vertex number
    - The program will check if the vertex exists
        - if it doesn’t then we will return ***-1***
        - Otherwise, we will return a Vertex function, which will return the in degree/ out degree
- iterateThroughOutBoundVertex, iterateThroughInBoundVertex – will receive a number, which represents the vertex number
    - The program will check if the vertex exists
        - if it doens’t we return ***None***
        - Otherwise, we will *yield* every element which has a connection for the vertex
- editCost – will get a new cost, starting point and the ending point for the vertex
    - Will check if the edge exists
        - if it doesn’t, we return None
    - Otherwise, we change the cost and we return 1
- getCost – will receive a starting point and an ending point
    - If there is no edge, we return -1
    - Otherwise we will return the cost of the edge

Vertex class functions:

- *getters for every class member*
- *increaseInDegree* – will increase the in degree of a vertex

Edge class functions:

- *getters for every class member*
- *setCost* – will receive an integer, which represents the new cost of the edge
    - will change the new cost of the edge