Specification

We define a class named DirectedGraph representing a directed graph.

The class DirectedGraph will provide the following methods:

__init__(vertices : int = 0, edges : int = 0)
Constructs a graph.

number_of_vertices()
Returns the number of vertices.

number_of_predecessors(vertex : int)
Returns the number of predecessors of a given vertex.

number_of_successors(vertex : int)
Returns the number of successors of a given vertex.

number_of_edges()
Returns the number of edges.

vertices_iterator()
Yields every vertex (as an int) in the graph

predecessors_iterator(vertex : int)
Yields every predecessor (as an int) of a given vertex

successors_iterator(vertex : int)
Yields every successor (as an int) of a given vertex.

edges_iterator()
Yields every edge (as a tuple (vertex1, vertex2, cost)) in the graph.

is_vertex(vertex : int)
Returns True if the vertex exists, False otherwise

is_edge(vertex1 : int, vertex2 : int)
Returns True if the edge vertex1 – vertex2 exists, False otherwise.

edge_cost(vertex1 : int, vertex2 : int)
Returns the cost of the edge vertex1 – vertex2.
If the edge doesn’t exist, it raises an error.

set_edge_cost(vertex1 : int, vertex2 : int, new_cost : int)
Sets the cost of the edge vertex1 – vertex2 to new_cost.
If the edge doesn’t exist, it raises an error.

add_vertex(vertex : int)
Adds a vertex to the graph.
If the vertex already exists, it raises an error.

add_edge(vertex1 : int, vertex2 : int, cost : int = 0)

Adds an edge to the graph.

If the edge already exists, it raises an error.

If one of the vertices does not exist, an error is raised.

remove_vertex(vertex : int)

Removes a vertex.

If the vertex doesn’t exist, it raises an error.

remove_edge(vertex1: int, vertex2 : int)

Removes an edge.

If the edge doesn’t exist, it raises an error.

copy()

Returns a copy of the graph

Implementation

The implementation uses a set of integers to store the vertices, two dictionaries of (key, values) as (integers, sets) for storing the predecessors and successors and a dictionary of (key, values) as (pair of integers, integer) for storing the edges.