graphs_homework.graph

▶ View Source

class Graph: ▶ View Source

Graph(nr_of_vertices=0, nr_of_edges=0) ▶ View Source

Creates a graph with the given number of vertices and edges.

def get_nr_of_vertices(self): ▶ View Source

Returns the number of vertices in the graph.

def vertices_iterator(self): ▶ View Source

Returns an iterator for the vertices in the graph.

def is_edge(self, vertex_from, vertex_to): ▶ View Source

Returns True if there is an edge from vertex_from to vertex_to, False otherwise.

def get_in_degree(self, vertex_to): ▶ View Source

Returns the in degree of a vertex.

def get_out_degree(self, vertex_from): ▶ View Source

Returns the out degree of a vertex.

def outbound_iterator(self, vertex_from): ▶ View Source

Returns an iterator for the outbound edges of a vertex.

def inbound_iterator(self, vertex_to): ▶ View Source

Returns an iterator for the inbound edges of a vertex.

def edges_iterator(self): ▶ View Source

Returns an iterator for the edges in the graph. Each edge is represented as a tuple (vertex_from, vertex_to).

def get_cost(self, vertex_from, vertex_to): ▶ View Source

Returns the cost of an edge. Throws an exception if the edge does not exist.

def set_cost(self, vertex_from, vertex_to, new_cost): ▶ View Source

Sets the cost of an edge. Throws an exception if the edge does not exist.

def add_vertex(self, vertex_to_be_added): ▶ View Source

Adds a vertex to the graph. Throws an exception if the vertex already exists.

def remove_vertex(self, vertex_to_be_removed): ▶ View Source

Removes a vertex from the graph. Throws an exception if the vertex does not exist.

def add_edge(self, vertex_from, vertex_to, edge_cost): ▶ View Source

Adds an edge to the graph. Throws an exception if the edge already exists or if one of the vertices does not exist.

def remove_edge(self, vertex_from, vertex_to): ▶ View Source

Removes an edge from the graph. Throws an exception if the edge does not exist.

def copy(self): ▶ View Source

Returns a copy of the graph.

def is_vertex(self, vertex_to_be_checked): ▶ View Source

Returns True if the vertex_to_be_checked is a vertex in the graph, False otherwise.

def get_nr_of_edges(self): ▶ View Source

Returns the number of edges in the graph.

graphs_homework.operations

▶ View Source

def read_graph_from_file(file_path): ▶ View Source

Reads a graph from a file.

def save_graph_to_file(graph, file_path): ▶ View Source

Saves a graph to a file.

def generate_random_graph(nr_of_vertices, nr_of_edges): ▶ View Source

Generates a random graph with the given number of vertices and edges.