Graphs lab 4 Documentation

Rapeanu George - Alexandru

CONTENTS:

1 python						
	1.1	UI module	1			
	1.2	UndirectedGraph module	1			
	1.3	UndirectedGraphTests module	4			
2 Indices and tables						
Python Module Index						
In	dex		11			

CHAPTER

ONE

PYTHON

1.1 UI module

```
UI.display_edges (edges)
This function displays a given list of edges

Parameters edges (list) - list of edges represented as tuples

Returns None

UI.display_graph (graph)
displays a graph

Parameters graph (UndirectedGraph.UndirectedGraph) - the graph
Returns None

UI.display_vertices (vertices)
This function displays the given vertices

Parameters vertices (list) - the vertices

Returns None

UI.main()
The main of the program

Returns None
```

1.2 UndirectedGraph module

```
class UndirectedGraph.UndirectedGraph (vertices, edges)
    Bases: object

add_edge (x, y, z)
    This function adds the edge from x to y to the graph

Parameters

• x (str) - the first vertex

• y (str) - the second vertex

• z (int) - the cost
```

Raises

 \bullet Exception-if types do not follow the specification

```
• Exception – if nodes do not exist
```

• Exception – if edge already exists

$add_vertex(x)$

This function adds the vertex x to the graph

Parameters \mathbf{x} (str) – the vertex to be added

Raises

- Exception if x is not string
- Exception if x already exists

copy()

This function retrieves a copy of the current graph

Returns a Graph copy

get_degree (x)

This function returns the degree of a vertex

Parameters \mathbf{x} (str) – the vertex

Returns the in degree of the vertex x

Raises Exception – if x doesn't exist

$get_edge_cost(x, y)$

This function returns the cost of the edge between x and y

Parameters

- $\mathbf{x}(str)$ the first vertex
- $\mathbf{y}(str)$ the second vertex

Returns the cost of the edge from x to y

Raises Exception – if there is no edge from x to y

has_vertex(vertex)

This function returns true if the provided vertex exists, false otherwise

Parameters vertex (str) - the vertex

Returns boolean

$is_edge(x, y)$

This function returns True if the edge x-y exists, false otherwise

Parameters

- \mathbf{x} (str) the first vertex
- $\mathbf{y}(str)$ the second vertex

Returns True if an edge exists, false otherwise

Raises Exception – if x or y are not vertices

$modify_edge_cost(x, y, z)$

This function modifies the cost of the edge from x to y

Parameters

• $\mathbf{x}(str)$ – the first vertex

2 Chapter 1. python

- **y** (str) the second vertex
- **z** (int) the new cost

Raises Exception – if there is no edge from x to y

parse_adjacent_edges (x)

This function returns an iterable of deepcopied vertices

Parameters \mathbf{x} – the vertex for which to retrieve the iterator

Returns iterator to a deepcopied list of outbound vertices

Raises Exception - if the vertex doesn't exist

parse_vertices()

This function returns an iterable containing nodes

The nodes are deepcopied, in order to avoid being modified from the outside :return: iterator through a list of deepcopied nodes

$remove_edge(x, y)$

This function removes the edge from x to y from the graph

Parameters

- \mathbf{x} (str) the first vertex
- $\mathbf{y}(str)$ the second vertex

Raises Exception – if edge already exists

$remove_vertex(x)$

This function removes the vertex x from the graph

Parameters \mathbf{x} (str) – the vertex to be removed

Raises Exception – if x doesn't exist

UndirectedGraph.get_connected_components(graph)

Returns a list of UndirectedGraph-s representing the connected component of the given graph

Parameters graph (UndirectedGraph) - the graph

Returns list of UndirectedGraph

UndirectedGraph.get_minimum_spanning_tree(graph)

Given a graph, this function returns the MST of it using Kruskal's algorithm

Parameters graph (UndirectedGraph) - the graph

Returns the tree as an UndirectedGraph

UndirectedGraph.random_graph(n, m)

This function creates a random graph with specified number of vertices and edges

Parameters

- n (int) the number of vertices
- m (int) the number of edges

Returns a graph with specified parameters

Raises Exception – if invalid parameters

UndirectedGraph.read graph(filename)

This function reads a graph from a file. It supports 2 formats .txt and .modified.txt

In case of .txt, the file is supposed to look like this:

On the first line, the number n of vertices and the number m of edges; On each of the following m lines, three numbers, x, y and c, describing an edge.

In case of .modified.txt, the file is supposed to look like this:

On the first line, the number n of vertices and the number m of edges On the second line, a list of the n vertices separated by space On each of the following m lines, three numbers, x, y and c, describing an edge.

Parameters filename (str) – the file from which to read(name, relative path or absolute path)

Returns Graph

Raises Exception - in case of invalid format

UndirectedGraph.write_graph (filename, graph)

This function writes a graph from a file. It supports 1 format .modified.txt

On the first line, the number n of vertices and the number m of edges On the second line, a list of the n vertices separated by space On each of the following m lines, three numbers, x, y and c, describing an edge.

Parameters

- filename (str) the filename to which to read(name, relative path or absolute path),
 MUST end in .modified.txt
- graph (UndirectedGraph) the graph to be written

Raises Exception – if invalid data

1.3 UndirectedGraphTests module

```
class UndirectedGraphTests.UndirectedGraphTests (methodName='runTest')
Bases: unittest.case.TestCase
setUp()
    Hook method for setting up the test fixture before exercising it.

test_add_edge()
test_add_vertex()
test_constructor()
test_copy()
test_eq()
test_get_MST()
test_get_connected_components()
test_get_degree()
test_get_edge_cost()
test_has_vertex()
```

4 Chapter 1. python

```
test_is_edge()
test_modify_edge_cost()
test_parse_adjacent_edges()
test_parse_vertices()
test_random_graph()
test_read_graph()
test_remove_edge()
test_remove_vertex()
test_write_graph()
```

6 Chapter 1. python

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

u

UI, 1 UndirectedGraph, 1 UndirectedGraphTests, 4

10 Python Module Index

INDEX

A add_edge() (UndirectedGraph.UndirectedGraph	$parse_vertices() \qquad \qquad (\textit{Undirected-}\\ \textit{Graph.UndirectedGraph method}), 3$
method), 1 add_vertex() (UndirectedGraph.UndirectedGraph method), 2 C copy() (UndirectedGraph.UndirectedGraph method), 2	R random_graph() (in module UndirectedGraph), 3 read_graph() (in module UndirectedGraph), 3 remove_edge() (UndirectedGraph.UndirectedGraph
D display_edges() (in module UI), 1	Graph. UndirectedGraph method), 3
<pre>display_graph() (in module UI), 1 display_vertices() (in module UI), 1</pre>	setUp() (UndirectedGraphT- ests.UndirectedGraphTests method), 4
<pre>G get_connected_components() (in module Undi-</pre>	T test_add_edge() (UndirectedGraphT- ests.UndirectedGraphTests method), 4 test_add_vertex() (UndirectedGraphT- ests.UndirectedGraphTests method), 4 test_constructor() (UndirectedGraphT- ests.UndirectedGraphTests method), 4 test_copy() (UndirectedGraphT- ests.UndirectedGraphTests method), 4 test_eq() (UndirectedGraphT- ests.UndirectedGraphTests method), 4 test_get_connected_components() (Undirect- edGraphTests.UndirectedGraphTests method), 4
is_edge() (UndirectedGraph.UndirectedGraph method), 2	test_get_degree() (UndirectedGraphT-ests.UndirectedGraphTests method), 4 test_get_edge_cost() (UndirectedGraphT-
M main() (in module UI), 1 modify_edge_cost() (Undirected- Graph.UndirectedGraph method), 2 P	ests.UndirectedGraphTests method), 4 test_get_MST() (UndirectedGraphTests.UndirectedGraphTests method), 4 test_has_vertex() (UndirectedGraphTests.UndirectedGraphTests method), 4 test_is_edge() (UndirectedGraphTests.U
$parse_adjacent_edges() \qquad \qquad (\textit{Undirected-Graph.UndirectedGraph method}), 3$	test_modify_edge_cost() (UndirectedGraphT-ests.UndirectedGraphTests method), 5

```
test_parse_adjacent_edges()
                                      (Undirected-
        GraphTests.UndirectedGraphTests
                                         method),
test_parse_vertices()
                               (UndirectedGraphT-
        ests.UndirectedGraphTests method), 5
test_random_graph()
                               (UndirectedGraphT-
        ests. UndirectedGraphTests method), 5
test_read_graph()
                               (UndirectedGraphT-
        ests. UndirectedGraphTests method), 5
test_remove_edge()
                               (UndirectedGraphT-
        ests. Undirected Graph Tests method), 5
test_remove_vertex()
                               (UndirectedGraphT-
        ests. Undirected Graph Tests method), 5
test_write_graph()
                               (UndirectedGraphT-
        ests. Undirected Graph Tests method), 5
U
UI (module), 1
UndirectedGraph (class in UndirectedGraph), 1
UndirectedGraph (module), 1
UndirectedGraphTests (class in Undirected-
        GraphTests), 4
UndirectedGraphTests (module), 4
W
write_graph() (in module UndirectedGraph), 4
```

12 Index