МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Дніпровський національний університет
залізничного транспорту імені академіка В. Лазаряна

Кафедра «Комп'ютерні інформаційні технології»

Лабораторна робота №3

з дисципліни «Професійна практика програмної інженерії»

на тему: “Автоматична документація”

Виконали:
студенти гр.ПЗ1911
Сафонов Д.Є.
Дорогокупля К.О.
Прийняла:
Шаповал І.В.

Дніпро, 2021

Table of Contents

Hierarchical Index

Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

- AbstractGraph
  - simple_graph_library::AdjacencyListGraph< T >....4
- simple_graph_library::IGraph< T >....5
- simple_graph_library::Vertex< T >....7

Class Index

Class List

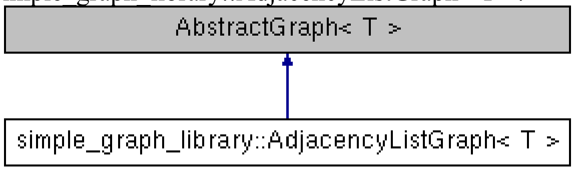
Here are the classes, structs, unions and interfaces with brief descriptions:

Class Documentation

simple_graph_library::AdjacencyListGraph< T > Class Template Reference

`#include <adjacency_list_graph.h>`

Inheritance diagram for simple_graph_library::AdjacencyListGraph< T >:



Public Member Functions

std::vector< **Vertex**< T > > ADJACENCY_LIST_GRAPH_EXPORT_FUNC **neighbours** (**Vertex**< T > v) final

IGraph<T> override, check the description in that class.

void ADJACENCY_LIST_GRAPH_EXPORT_FUNC **add_vertex** (**Vertex**< T > v) final

IGraph<T> override, check the description in that class.

void ADJACENCY_LIST_GRAPH_EXPORT_FUNC **add_edge** (**Vertex**< T > a, **Vertex**< T > b) final

IGraph<T> override, check the description in that class.

void ADJACENCY_LIST_GRAPH_EXPORT_FUNC **remove_vertex** (**Vertex**< T > v) final

IGraph<T> override, check the description in that class.

void ADJACENCY_LIST_GRAPH_EXPORT_FUNC **remove_edge** (**Vertex**< T > a, **Vertex**< T > b) final

IGraph<T> override, check the description in that class.

Detailed Description

template<class T>

class simple_graph_library::AdjacencyListGraph< T >

Representation of Graph with Adjacency List.

The documentation for this class was generated from the following file:

/home/dazzlemon/hdd_backup/github/diit_121-ipz_y2_pppi/2/simple_graph_library/lib/src/main/public/adjacency_list_graph.h

simple_graph_library::IGraph< T > Class Template Reference

Public Member Functions

virtual auto **dfs** (**Vertex**< T > v) -> std::vector< **Vertex**< T >> IGRAPH_EXPORT_FUNC=0
virtual auto **bfs** (**Vertex**< T > v) -> std::vector< **Vertex**< T >> IGRAPH_EXPORT_FUNC=0
virtual auto **euler_tour** () -> std::vector< **Vertex**< T >> IGRAPH_EXPORT_FUNC=0
virtual auto **neighbours** (**Vertex**< T > v) -> std::vector< **Vertex**< T >> IGRAPH_EXPORT_FUNC=0

Returns vector of vertices that have common edge with <v>.

virtual void IGRAPH_EXPORT_FUNC **add_vertex** (**Vertex**< T > v)=0

Adds vertex <v> to this graph.

virtual void IGRAPH_EXPORT_FUNC **add_edge** (**Vertex**< T > a, **Vertex**< T > b)=0

Adds edge between two vertices and **.**

virtual void IGRAPH_EXPORT_FUNC **remove_vertex** (**Vertex**< T > v)=0

Removes vertex <v> from this graph.

virtual void IGRAPH_EXPORT_FUNC **remove_edge** (**Vertex**< T > a, **Vertex**< T > b)=0

Removes edge between two vertices.

Member Function Documentation

template<class T > virtual void IGRAPH_EXPORT_FUNC simple_graph_library::IGraph< T >::add_edge (Vertex< T > *a*, Vertex< T > *b*)[pure virtual]

Adds edge between two vertices and **.**

Parameters

in	*a*	first vertex for new common edge
in	*b*	second vertex for new common edge

if either or **is not in the graph edge is not added.**

template<class T > virtual void IGRAPH_EXPORT_FUNC simple_graph_library::IGraph< T >::add_vertex (Vertex< T > *v*)[pure virtual]

Adds vertex <v> to this graph.

Parameters

in	*v*	vertex to add

Vertex doesnt have any edges after adding, they have to be declared separately by function add_edge.

template<class T > virtual auto simple_graph_library::IGraph< T >::neighbours (Vertex< T > *v*) -> std::vector< Vertex< T >> IGRAPH_EXPORT_FUNC `[pure virtual]`

Returns vector of vertices that have common edge with <v>.

Parameters

in	*v*	vertex to get neighbours from

Returns

vector of neighbouring vertices to <v>

template<class T > virtual void IGRAPH_EXPORT_FUNC simple_graph_library::IGraph< T >::remove_edge (Vertex< T > *a*, Vertex< T > *b*) `[pure virtual]`

Removes edge between two vertices.

Parameters

in	*a*	first vertex to remove common edge from
in	*b*	second vertex to remove common edge from

if either or **is not in the graph or they dont common edge its not removed(obviously).**

template<class T > virtual void IGRAPH_EXPORT_FUNC simple_graph_library::IGraph< T >::remove_vertex (Vertex< T > *v*)`[pure virtual]`

Removes vertex <v> from this graph.

Parameters

in	*v*	vertex to remove

if <v> is not in the graph its not removed(obviously).

The documentation for this class was generated from the following file:

/home/dazzlemon/hdd_backup/github/diit_121-ipz_y2_pppi/2/simple_graph_library/lib/src/main/public/igraph.h

simple_graph_library::Vertex< T > Class Template Reference

```
#include <vertex.h>
```

Public Member Functions

Vertex (const T &data)

Public Attributes

T **data**

Friends

auto **operator<=>** (const **Vertex** &lhs, const **Vertex**< T > &rhs) -> std::strong_ordering

---

Detailed Description

template<class T>

class simple_graph_library::Vertex< T >

Just a wrapper for T dataclass, T includes both Key&Value

The documentation for this class was generated from the following file:

/home/dazzlemon/hdd_backup/github/diit_121-ipz_y2_pppi/2/simple_graph_library/lib/src/main/public/vertex.h

Index