Class Graph

Generated by Doxygen 1.8.13

Contents

Chapter 1

Класс Graph

1.1 Введение

Класс Graph предназначен для работы с неориентированными, связными графами.

Ссылка на документацию

Chapter 2

graph

```
flowchart LR;
   1-- 1 ---2;
   1-- 7 ---3;
   1-- 7 ---4;
   2-- 1 ---4;
   3-- 1 ---4;
```

Chapter 3

Graph

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

Class Documentation

5.1 Edge Class Reference

Класс Edge реализует ребра графа.

#include <graph.h>

Public Member Functions

- Edge (int from_v, int to_v)
- Edge (int from_v, int to_v, int weight)

Public Attributes

- int from_vertex
- int other_vertex
- int weight = 1

5.1.1 Detailed Description

Класс Edge реализует ребра графа.

Каждый объект класса Edge хранит в себе следующую информацию: from_vertex - одна из вершин ребра other_vertex - другая вершина ребра weight - вес ребра

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

- source/graph/graph.h

5.2 Graph Class Reference

Класс Graph основной класс реализующий граф, поддерживающий добавление и удаление вершин и ребер, то есть способный динамически изменяться.

#include <graph.h>

Public Member Functions

- Graph (const std::vector< Edge > &edge)
- Graph (const Graph &other)
- Graph & operator= (const Graph &other)
- Graph (Graph &&other) noexcept
- Graph & operator= (Graph &&other) noexcept
- void AddVertex (const int &v_num)
- void AddVertex (const int &v_num, std::vector< int > &edges)
- void AddVertex (const int &v_num, const std::vector< int > &edges, const std::vector< int > &weights)
- void RemoveVertex (const int &v_num)
- void AddEdge (const Edge &new_edge)
- void AddEdge (const int &from_v, const int &to_v)
- void AddEdge (const int &from_v, const int &to_v, const int &weight)
- void RemoveEdge (const int &from_v, const int &to_v)
- Graph FindMST ()
- int Size ()
- std::vector< Edge > AllEdges ()
- std::vector< int > AllVertex ()
- void ShowGraph (std::string file_name) const
- std::istream & ReadFrom (std::istream &)
- std::ostream & WriteTo (std::ostream &) const

5.2.1 Detailed Description

Класс Graph основной класс реализующий граф, поддерживающий добавление и удаление вершин и ребер, то есть способный динамически изменяться.

Каждый объект класса Graph хранит в себе следующую информацию: vertex - std::vector<int> с вершинами графа edge - std::vector<std::vector<Edge>> матрица смежности графа, то есть множество ребер графа

5.2.2 Constructor & Destructor Documentation

5.2.2.1 Graph()

Graph::Graph (
const std::vector< Edge > & edge)

Создает объект класса Graph

Parameters

edge	список ребер, которые задают граф

5.2.3 Member Function Documentation

5.2.3.1 AddEdge() [1/3]

```
void Graph::AddEdge (
        const Edge & new_edge )
```

Добавляет указанное ребро в граф

Parameters

new_edge	ребро, которое нужно добавить в граф

Exceptions

std::exception	Если ребро, которое хотим добавить, уже есть в графе

5.2.3.2 AddEdge() [2/3]

```
void Graph::AddEdge (
        const int & from_v,
        const int & to_v )
```

Добавляет в граф ребро между вершинами

Parameters

from↩ _v	одна из вершин ребра
to_v	другая вершина ребра

Exceptions

std::exception	Если ребро, которое хотим добавить, уже есть в графе

5.2.3.3 AddEdge() [3/3]

```
void Graph::AddEdge (
        const int & from_v,
        const int & to_v,
        const int & weight )
```

Добавляет в граф ребро с указанным весом между вершинами

Parameters

from↩ _v	одна из вершин ребра
to_v	другая вершина ребра
weight	вес ребра

Exceptions

std::exception	Если ребро, которое хотим добавить, уже есть в графе

5.2.3.4 AddVertex() [1/3]

```
void Graph::AddVertex (
            const int & v_num )
```

Добавляет одну вершину в граф, обычно используется при добавлении первой вершины

Parameters

v_num	номер вершины

Exceptions

std::exception	Если вершина, которую хотим добавить, уже есть в графе

5.2.3.5 AddVertex() [2/3]

```
void Graph::AddVertex (
            const int & v_num,
            std::vector< int > & edges )
```

Добавляет в граф вершину и её ребра. Необходимо, чтобы все вершины в списке уже находились в графе

Parameters

v_num	номер вершины
edges	список смежных вершин

Exceptions

std::exception	Если вершина, которую хотим добавить, уже есть в графе

5.2.3.6 AddVertex() [3/3]

void Graph::AddVertex (
const int & v_num,
const std::vector< int > & edges,
const std::vector< int > & weights)

Добавляет в граф вершину и её ребра с весами. Необходимо, чтобы все вершины в списке уже находились в графе

Parameters

v_num	номер вершины
edges	список смежных вершин
weights	список весов ребер

Exceptions

std::exception	Если вершина, которую хотим добавить, уже есть в графе

5.2.3.7 AllEdges()

std::vector< Edge > Graph::AllEdges ( )

Функция, показывающая ребра графа

Returns

Список всех ребер графа

5.2.3.8 AllVertex()

std::vector< int > Graph::AllVertex ( )

Функция, показывающая вершины графа

Returns

Список всех вершин графа

5.2.3.9 FindMST()

Graph Graph::FindMST ( )

Функция поиска минимального остовного дерева в связном, взвешенном графе

Returns

Объект класса Graph, являющийся минимальным остовным деревом исходного графа

Exceptions

std::exception	Если на вход был подан некорректный граф, для которого нельзя построить минимальное остовное дерево

5.2.3.10 RemoveEdge()

```
void Graph::RemoveEdge (
            const int & from_v,
            const int & to_v )
```

Удаляет ребро между двумя вершинами

Parameters

from↩ _v	одна из вершин ребра
to_v	другая вершина ребра

Exceptions

std::exception	Если в графе нет ребра, которое хотим удалить

5.2.3.11 RemoveVertex()

```
void Graph::RemoveVertex (
            const int & v_num )
```

Удаляет вершину из графа вместе со всеми её ребрами

Parameters

v_num	номер вершины

Exceptions

std::exception	Если в графе нет вершины, которую хотим удалить

5.2.3.12 ShowGraph()

```
void Graph::ShowGraph (
            std::string file_name ) const
```

Визуализирует граф

Parameters

file_name	имя .md файла, в которое будет записано описание графа

Note

После компиляции .md файла будет получено визуальное представление графа

5.2.3.13 Size()

int Graph::Size ( )

Функция определения размера графа

Returns

Количество вершин в графе

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

- source/graph/graph.h
- source/graph/findMST.cpp
- source/graph/graph.cpp

5.3 Set Class Reference

Вспомогательный класс Set непересекающихся множеств для эффективной реализации алгоритма Краскала.

Public Member Functions

- void MakeSet (const size_t &ind)
- void Union (Set *other)
- Set * FindSet ()

5.3.1 Detailed Description

Вспомогательный класс Set непересекающихся множеств для эффективной реализации алгоритма Краскала.

Каждый объект класса Set хранит в себе следующую информацию: root - указатель на Set, который является корневым для поддерева rank - высота поддерева

5.3.2 Member Function Documentation

5.3.2.1 FindSet()

Set* Set::FindSet ( ) [inline]

Определяет к какому множеству относится поддерево

Returns

Возвращает указатель на объект класса Set, который является корневым для поддерева

5.3.2.2 MakeSet()

void Set::MakeSet (
const size_t & ind) [inline]

Создает объект класса Set для одной вершины

Parameters

ind	номер вершины

5.3.2.3 Union()

void Set::Union (
Set * other) [inline]

Объединяет множество с другим

Parameters

other	указатель на другой объект класса Set

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

- source/graph/findMST.cpp

Index