

Graf nieskierowany

Wygenerowano przez Doxygen 1.8.4

N, 11 maj 2014 23:45:56

Spis treści

1	Graf nieskierowany z wagami	1
1.1	Opis programu	1
1.2	Autor	1
2	Indeks klas	2
2.1	Lista klas	2
3	Indeks plików	2
3.1	Lista plików	2
4	Dokumentacja klas	2
4.1	Dokumentacja struktury <code>Graph< Type >::Edge</code>	2
4.1.1	Dokumentacja konstruktora i destruktoru	2
4.1.2	Dokumentacja atrybutów składowych	2
4.2	Dokumentacja szablonu klasy <code>Graph< Type ></code>	3
4.2.1	Opis szczegółowy	3
4.2.2	Dokumentacja składowych definicji typu	3
4.2.3	Dokumentacja funkcji składowych	3
4.2.4	Dokumentacja atrybutów składowych	4
5	Dokumentacja plików	5
5.1	Dokumentacja pliku <code>graph.hh</code>	5
5.1.1	Opis szczegółowy	5
5.2	Dokumentacja pliku <code>main.cpp</code>	5
5.2.1	Opis szczegółowy	5
5.2.2	Dokumentacja funkcji	5
5.3	Dokumentacja pliku <code>mainpage.dox</code>	5
	Indeks	6

1 Graf nieskierowany z wagami

1.1 Opis programu

Program definiuje strukturę danych do reprezentacji obiektów, pomiędzy którymi występują różne zależności. Graf składa się z liczby wierzchołków oraz k liczby krawędzi - w przypadku implementowanego grafu (nieskierowanego) wierzchołki można łączyć w obie strony.

1.2 Autor

Program wykonała: Agnieszka Wisniewska, nr albumu: 200 466

2 Indeks klas

2.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

Graph< Type >::Edge	2
Graph< Type >	
Class Graph Jest to klasa definiująca graf nieskierowany z wagą pozwalająca na wykonywaniu wybranych funkcji	3

3 Indeks plików

3.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

graph.hh	5
main.cpp	
Plik zawierający główną funkcję programu	5

4 Dokumentacja klas

4.1 Dokumentacja struktury Graph< Type >::Edge

```
#include <graph.hh>
```

Metody publiczne

- [Edge](#) (const Type newEnd, const int newWeight)

Atrybuty publiczne

- Type [SecEnd](#)
- int [Weight](#)

4.1.1 Dokumentacja konstruktora i destruktora

```
4.1.1.1 template<typename Type> Graph< Type >::Edge::Edge ( const Type newEnd, const int newWeight )  
[inline]
```

4.1.2 Dokumentacja atrybutów składowych

```
4.1.2.1 template<typename Type> Type Graph< Type >::Edge::SecEnd
```

```
4.1.2.2 template<typename Type> int Graph< Type >::Edge::Weight
```

Dokumentacja dla tej struktury została wygenerowana z pliku:

- [graph.hh](#)

4.2 Dokumentacja szablonu klasy Graph< Type >

class **Graph** Jest to klasa definiująca graf nieskierowany z wagą pozwalająca na wykonywaniu wybranych funkcji.

```
#include <graph.hh>
```

Komponenty

- struct **Edge**

Typy publiczne

- typedef std::vector< **Edge** > **EdgeS**

Metody publiczne

- void **AddVert** (const Type &vert)
Funkcja dodająca nowy wierzchołek.
- void **RemoveVert** (const Type &vert)
Funkcja usuwająca wybrany wierzchołek.
- void **AddEdge** (const Type &vert1, const Type &vert2, const int Weight=0)
Funkcja dodająca nową krawędź
- void **RemoveEdge** (const Type &vert1, const Type &vert2)
Funkcja usuwająca daną krawędź
- bool **IfConnected** (const Type &vert1, const Type &vert2)
Funkcja sprawdzająca, czy wierzchołki są ze sobą bezpośrednio połączone.
- **EdgeS Neighbors** (const Type &vert)
Funkcja znajdujące sąsiednie wierzchołki.

Atrybuty prywatne

- std::map< Type, **EdgeS** > **graph**

4.2.1 Opis szczegółowy

```
template<typename Type>class Graph< Type >
```

class **Graph** Jest to klasa definiująca graf nieskierowany z wagą pozwalająca na wykonywaniu wybranych funkcji.

4.2.2 Dokumentacja składowych definicji typu

4.2.2.1 template<typename Type> typedef std::vector<Edge> Graph< Type >::EdgeS

4.2.3 Dokumentacja funkcji składowych

4.2.3.1 template<typename Type> void Graph< Type >::AddEdge (const Type & vert1, const Type & vert2, const int Weight = 0)

Funkcja dodająca nową krawędź

Parametry

<i>vert1</i>	współrzędna pierwszego wierzchołka
<i>vert2</i>	współrzędna drugiego wierzchołka
<i>Weight</i>	waga krawędzi

4.2.3.2 `template<typename Type> void Graph< Type >::AddVert (const Type & vert)`

Funkcja dodająca nowy wierzchołek.

Parametry

<i>vert</i>	wartość dodawanego wierzchołka
-------------	--------------------------------

4.2.3.3 `template<typename Type> bool Graph< Type >::IfConnected (const Type & vert1, const Type & vert2)`

Funkcja sprawdzająca, czy wierzchołki są ze sobą bezpośrednio połączone.

Parametry

<i>vert1</i>	współrzędna pierwszego wierzchołka
<i>vert2</i>	współrzędna drugiego wierzchołka

Zwraca

true jeśli wierzchołki są połączone
false jeśli wierzchołki nie są połączone

4.2.3.4 `template<typename Type> Graph< Type >::EdgeS Graph< Type >::Neighbors (const Type & vert)`

Funkcja znajdujące sąsiednie wierzchołki.

Parametry

<i>vert</i>	wierzchołek, którego sąsiadów poszukujemy
-------------	---

Zwraca

wektor wierzchołków sąsiadujących

4.2.3.5 `template<typename Type> void Graph< Type >::RemoveEdge (const Type & vert1, const Type & vert2)`

Funkcja usuwająca daną krawędź

Parametry

<i>vert1</i>	współrzędna pierwszego wierzchołka
<i>vert2</i>	współrzędna drugiego wierzchołka

4.2.3.6 `template<typename Type> void Graph< Type >::RemoveVert (const Type & vert)`

Funkcja usuwająca wybrany wierzchołek.

Parametry

<i>vert</i>	wartość usuwanego wierzchołka
-------------	-------------------------------

4.2.4 Dokumentacja atrybutów składowych

4.2.4.1 `template<typename Type> std::map< Type, EdgeS > Graph< Type >::graph [private]`

Dokumentacja dla tej klasy została wygenerowana z pliku:

- [graph.hh](#)

5 Dokumentacja plików

5.1 Dokumentacja pliku graph.hh

```
#include <vector>
#include <map>
```

Komponenty

- class [Graph< Type >](#)
class [Graph](#) Jest to klasa definiująca graf nieskierowany z wagą pozwalającą na wykonywaniu wybranych funkcji.
- struct [Graph< Type >::Edge](#)

5.1.1 Opis szczegółowy

Plik zawierający szablon klasy [Graph](#).

5.2 Dokumentacja pliku main.cpp

Plik zawierający główną funkcję programu.

```
#include "graph.hh"
#include <iostream>
#include <cstdlib>
#include <ctime>
```

Funkcje

- int [main](#) (int argc, char **argv)
Główna funkcja programu.

5.2.1 Opis szczegółowy

Plik zawierający główną funkcję programu.

5.2.2 Dokumentacja funkcji

5.2.2.1 int main (int argc, char ** argv)

Główna funkcja programu.

5.3 Dokumentacja pliku mainpage.dox

Skorowidz

AddEdge

Graph, [3](#)

AddVert

Graph, [4](#)

Edge

Graph::Edge, [2](#)

EdgeS

Graph, [3](#)

Graph

AddEdge, [3](#)

AddVert, [4](#)

EdgeS, [3](#)

graph, [4](#)

IfConnected, [4](#)

Neighbors, [4](#)

RemoveEdge, [4](#)

RemoveVert, [4](#)

graph

Graph, [4](#)

Graph< Type >, [3](#)

Graph< Type >::Edge, [2](#)

graph.hh, [5](#)

Graph::Edge

Edge, [2](#)

SecEnd, [2](#)

Weight, [2](#)

IfConnected

Graph, [4](#)

main

main.cpp, [5](#)

main.cpp, [5](#)

main, [5](#)

mainpage.dox, [5](#)

Neighbors

Graph, [4](#)

RemoveEdge

Graph, [4](#)

RemoveVert

Graph, [4](#)

SecEnd

Graph::Edge, [2](#)

Weight

Graph::Edge, [2](#)