

PA10

Generated by Doxygen 1.8.11

Contents

1	Class Index	1
1.1	Class List	1
2	File Index	3
2.1	File List	3
3	Class Documentation	5
3.1	WeightedGraph::Vertex Class Reference	5
3.1.1	Member Function Documentation	5
3.1.1.1	getColor() const	5
3.1.1.2	getLabel() const	5
3.1.1.3	setColor(char newColor)	5
3.1.1.4	setLabel(const string &newLabel)	5
3.2	WeightedGraph Class Reference	5
3.2.1	Constructor & Destructor Documentation	6
3.2.1.1	WeightedGraph(int maxNumber=MAX_GRAPH_SIZE)	6
3.2.1.2	WeightedGraph(const WeightedGraph &other)	7
3.2.1.3	~WeightedGraph()	7
3.2.2	Member Function Documentation	7
3.2.2.1	areAllEven() const	7
3.2.2.2	clear()	8
3.2.2.3	getEdgeWeight(const string &v1, const string &v2, int &wt) const	8
3.2.2.4	hasProperColoring() const	8
3.2.2.5	insertEdge(const string &v1, const string &v2, int wt)	8

3.2.2.6	<code>insertVertex(const Vertex &newVertex)</code>	9
3.2.2.7	<code>isEmpty() const</code>	9
3.2.2.8	<code>isFull() const</code>	9
3.2.2.9	<code>operator=(const WeightedGraph &other)</code>	9
3.2.2.10	<code>removeEdge(const string &v1, const string &v2)</code>	10
3.2.2.11	<code>removeVertex(const string &v)</code>	10
3.2.2.12	<code>retrieveVertex(const string &v, Vertex &vData) const</code>	10
3.2.2.13	<code>showShortestPaths()</code>	11
3.2.2.14	<code>showStructure() const</code>	11
3.2.3	Member Data Documentation	11
3.2.3.1	<code>INFINITE_EDGE_WT</code>	11
3.2.3.2	<code>MAX_GRAPH_SIZE</code>	11
4	File Documentation	13
4.1	<code>config.h</code> File Reference	13
4.1.1	Macro Definition Documentation	13
4.1.1.1	<code>LAB12_TEST1</code>	13
4.1.1.2	<code>LAB12_TEST2</code>	13
4.1.1.3	<code>LAB12_TEST3</code>	13
4.2	<code>show12.cpp</code> File Reference	13
4.3	<code>test12.cpp</code> File Reference	13
4.3.1	Function Documentation	14
4.3.1.1	<code>main()</code>	14
4.3.1.2	<code>print_help()</code>	14
4.4	<code>WeightedGraph.cpp</code> File Reference	14
4.4.1	Detailed Description	14
4.5	<code>WeightedGraph.h</code> File Reference	14
	Index	15

Chapter 1

Class Index

1.1 Class List

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

WeightedGraph::Vertex	5
WeightedGraph	5

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

config.h	13
show12.cpp	13
test12.cpp	13
WeightedGraph.cpp	
An implementation of a Weighted Graph ADT	14
WeightedGraph.h	14

Chapter 3

Class Documentation

3.1 WeightedGraph::Vertex Class Reference

```
#include <WeightedGraph.h>
```

Public Member Functions

- void [setLabel](#) (const string &newLabel)
- string [getLabel](#) () const
- void [setColor](#) (char newColor)
- char [getColor](#) () const

3.1.1 Member Function Documentation

3.1.1.1 char WeightedGraph::Vertex::getColor () const [inline]

3.1.1.2 string WeightedGraph::Vertex::getLabel () const [inline]

3.1.1.3 void WeightedGraph::Vertex::setColor (char *newColor*) [inline]

3.1.1.4 void WeightedGraph::Vertex::setLabel (const string & *newLabel*) [inline]

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

- [WeightedGraph.h](#)

3.2 WeightedGraph Class Reference

```
#include <WeightedGraph.h>
```

Classes

- class [Vertex](#)

Public Member Functions

- [WeightedGraph](#) (int maxNumber=[MAX_GRAPH_SIZE](#))
"Constructor. Creates an empty graph. Allocates enough memory for a graph containing maxNumber vertices."
- [WeightedGraph](#) (const [WeightedGraph](#) &other)
"Copy Constructor. Initializes the weighted graph to be equivalent to the other weighted graph parameter."
- [WeightedGraph](#) & [operator=](#) (const [WeightedGraph](#) &other)
"Overloaded assignment operator. Sets the weighted graph to be equivalent to the other weighted graph parameter and returns a reference to this other."
- [~WeightedGraph](#) ()
"Destructor. Deallocates (frees) the memory used to store a graph."
- void [insertVertex](#) (const [Vertex](#) &newVertex) throw ([logic_error](#))
"Inserts newVertex into a graph. If the vertex already exists in the graph, then updates it."
- void [insertEdge](#) (const string &v1, const string &v2, int wt) throw ([logic_error](#))
"Inserts an undirected edge connecting vertices v1 and v2 into the graph. The weight of the edge is WT. If there is already an edge connecting these vertices, then updates the weight of the edge."
- bool [retrieveVertex](#) (const string &v, [Vertex](#) &vData) const
"Searches a graph for vertex v. If this vertex is found, then places the value of the vertex's data in vData and returns true. Otherwise, returns false with vData undefined."
- bool [getEdgeWeight](#) (const string &v1, const string &v2, int &wt) const throw ([logic_error](#))
"Searches the graph for the edge connecting vertices v1 and v2. If this edge exists, then places the weight of the edge in wt and returns true. Otherwise, returns false with wt undefined."
- void [removeVertex](#) (const string &v) throw ([logic_error](#))
"Removes vertex v from the graph and any edges connected to v."
- void [removeEdge](#) (const string &v1, const string &v2) throw ([logic_error](#))
"Removes the edge connecting vertices v1 and v2 from the graph."
- void [clear](#) ()
"Removes all the vertices and edges in the graph."
- bool [isEmpty](#) () const
"Returns true if the graph is empty (no vertices). Otherwise, returns false."
- bool [isFull](#) () const
"Returns true if the graph is full (cannot add any more vertices). Otherwise, returns false."
- void [showStructure](#) () const
- void [showShortestPaths](#) ()
"Computes and displays the graph's path matrix."
- bool [hasProperColoring](#) () const
"Returns true if no vertex in the graph has the same color as an adjacent vertex. Otherwise, returns false."
- bool [areAllEven](#) () const
"Returns true if every vertex in a graph is of even degree. Otherwise, returns false."

Static Public Attributes

- static const int [MAX_GRAPH_SIZE](#) = 10
- static const int [INFINITE_EDGE_WT](#) = INT_MAX

3.2.1 Constructor & Destructor Documentation

3.2.1.1 [WeightedGraph::WeightedGraph](#) (int maxNumber = [MAX_GRAPH_SIZE](#))

"Constructor. Creates an empty graph. Allocates enough memory for a graph containing maxNumber vertices."

Parameters

<i>int</i>	maxNumber
------------	-----------

Returns

none

3.2.1.2 WeightedGraph::WeightedGraph (const WeightedGraph & other)

"Copy Constructor. Initializes the weighted graph to be equivalent to the other weighted graph parameter."

Parameters

<i>const</i>	WeightedGraph & other
--------------	---------------------------------------

Returns

none

3.2.1.3 WeightedGraph::~~WeightedGraph ()

"Destructor. Deallocates (frees) the memory used to store a graph."

Parameters

<i>none</i>	
-------------	--

Returns

none

3.2.2 Member Function Documentation**3.2.2.1 bool WeightedGraph::areAllEven () const**

"Returns true if every vertex in a graph is of even degree. Otherwise, returns false."

Parameters

<i>none</i>	
-------------	--

Returns

true or false

3.2.2.2 void WeightedGraph::clear ()

"Removes all the vertices and edges in the graph."

Parameters

<i>none</i>	
-------------	--

Returns

none

3.2.2.3 bool WeightedGraph::getEdgeWeight (const string & v1, const string & v2, int & wt) const throw logic_error)

"Searches the graph for the edge connecting vertices v1 and v2. If this edge exists, then places the weight of the edge in wt and returns true. Otherwise, returns false with wt undefined."

Parameters

<i>const</i>	string& v1, const string& v2, int& wt
--------------	---------------------------------------

Returns**3.2.2.4 bool WeightedGraph::hasProperColoring () const**

"Returns true if no vertex in the graph has the same color as an adjacent vertex. Otherwise, returns false."

Parameters

<i>none</i>	
-------------	--

Returns

true or false

3.2.2.5 void WeightedGraph::insertEdge (const string & v1, const string & v2, int wt) throw logic_error)

"Inserts an undirected edge connecting vertices v1 and v2 into the graph. The weight of the edge is WT. If there is already an edge connecting these vertices, then updates the weight of the edge."

Parameters

<i>const</i>	string& v1, const string& v2, int wt
--------------	--------------------------------------

Returns

none

3.2.2.6 void WeightedGraph::insertVertex (const Vertex & newVertex) throw logic_error)

"Inserts newVertex into a graph. If the vertex already exists in the graph, then updates it."

Parameters

<i>newVertex</i>	
------------------	--

Returns

none

3.2.2.7 bool WeightedGraph::isEmpty () const

"Returns true if the graph is empty (no vertices). Otherwise, returns false."

Parameters

<i>none</i>	
-------------	--

Returns

true if empty, false if not

3.2.2.8 bool WeightedGraph::isFull () const

"Returns true if the graph is full (cannot add any more vertices). Otherwise, returns false."

Parameters

<i>none</i>	
-------------	--

Returns

true if full, false if not

3.2.2.9 WeightedGraph & WeightedGraph::operator= (const WeightedGraph & other)

"Overloaded assignment operator. Sets the weighted graph to be equivalent to the other weighted graph parameter and returns a reference to this other."

Parameters

<i>const</i>	WeightedGraph & other
--------------	---------------------------------------

Returns

*this

3.2.2.10 void `WeightedGraph::removeEdge (const string & v1, const string & v2)` throw `logic_error`)

"Removes the edge connecting vertices v1 and v2 from the graph."

Parameters

<i>const</i>	string& v1, const string& v2
--------------	------------------------------

Returns

none

3.2.2.11 void `WeightedGraph::removeVertex (const string & v)` throw `logic_error`)

"Removes vertex v from the graph and any edges connected to v."

Parameters

<i>const</i>	string& v
--------------	-----------

Returns

none

3.2.2.12 bool `WeightedGraph::retrieveVertex (const string & v, Vertex & vData)` const

"Searches a graph for vertex v. If this vertex is found, then places the value of the vertex's data in vData and returns true. Otherwise, returns false with vData undefined."

Parameters

<i>const</i>	string& v, Vertex & vData
--------------	---

Returns

true if found and false if not

3.2.2.13 void WeightedGraph::showShortestPaths ()

"Computes and displays the graph's path matrix."

Parameters

<i>none</i>	
-------------	--

Returns

none

3.2.2.14 void WeightedGraph::showStructure () const

3.2.3 Member Data Documentation

3.2.3.1 const int WeightedGraph::INFINITE_EDGE_WT = INT_MAX [static]

3.2.3.2 const int WeightedGraph::MAX_GRAPH_SIZE = 10 [static]

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

- [WeightedGraph.h](#)
- [show12.cpp](#)
- [WeightedGraph.cpp](#)

Chapter 4

File Documentation

4.1 config.h File Reference

Macros

- `#define LAB12_TEST1 1`
- `#define LAB12_TEST2 0`
- `#define LAB12_TEST3 1`

4.1.1 Macro Definition Documentation

4.1.1.1 `#define LAB12_TEST1 1`

[WeightedGraph](#) class configuration file. Activate test #N by defining the corresponding LAB12_TESTN to have the value 1.

4.1.1.2 `#define LAB12_TEST2 0`

4.1.1.3 `#define LAB12_TEST3 1`

4.2 show12.cpp File Reference

4.3 test12.cpp File Reference

```
#include <iostream>
#include <cstring>
#include <cctype>
#include "WeightedGraph.h"
#include "config.h"
```

Functions

- void [print_help](#) ()
- int [main](#) ()

4.3.1 Function Documentation

4.3.1.1 int main ()

4.3.1.2 void print_help ()

4.4 WeightedGraph.cpp File Reference

An implementation of a Weighted Graph ADT.

```
#include "WeightedGraph.h"  
#include "show12.cpp"
```

4.4.1 Detailed Description

An implementation of a Weighted Graph ADT.

Author

Christopher Eichstedt

4.5 WeightedGraph.h File Reference

```
#include <stdexcept>  
#include <iostream>  
#include <climits>  
#include <string>
```

Classes

- class [WeightedGraph](#)
- class [WeightedGraph::Vertex](#)

Index

- ~WeightedGraph
 - WeightedGraph, [7](#)
- areAllEven
 - WeightedGraph, [7](#)
- clear
 - WeightedGraph, [7](#)
- config.h, [13](#)
 - LAB12_TEST1, [13](#)
 - LAB12_TEST2, [13](#)
 - LAB12_TEST3, [13](#)
- getColor
 - WeightedGraph::Vertex, [5](#)
- getEdgeWeight
 - WeightedGraph, [8](#)
- getLabel
 - WeightedGraph::Vertex, [5](#)
- hasProperColoring
 - WeightedGraph, [8](#)
- INFINITE_EDGE_WT
 - WeightedGraph, [11](#)
- insertEdge
 - WeightedGraph, [8](#)
- insertVertex
 - WeightedGraph, [9](#)
- isEmpty
 - WeightedGraph, [9](#)
- isFull
 - WeightedGraph, [9](#)
- LAB12_TEST1
 - config.h, [13](#)
- LAB12_TEST2
 - config.h, [13](#)
- LAB12_TEST3
 - config.h, [13](#)
- MAX_GRAPH_SIZE
 - WeightedGraph, [11](#)
- main
 - test12.cpp, [14](#)
- operator=
 - WeightedGraph, [9](#)
- print_help
 - test12.cpp, [14](#)
- removeEdge
 - WeightedGraph, [10](#)
- removeVertex
 - WeightedGraph, [10](#)
- retrieveVertex
 - WeightedGraph, [10](#)
- setColor
 - WeightedGraph::Vertex, [5](#)
- setLabel
 - WeightedGraph::Vertex, [5](#)
- show12.cpp, [13](#)
- showShortestPaths
 - WeightedGraph, [10](#)
- showStructure
 - WeightedGraph, [11](#)
- test12.cpp, [13](#)
 - main, [14](#)
 - print_help, [14](#)
- WeightedGraph, [5](#)
 - ~WeightedGraph, [7](#)
 - areAllEven, [7](#)
 - clear, [7](#)
 - getEdgeWeight, [8](#)
 - hasProperColoring, [8](#)
 - INFINITE_EDGE_WT, [11](#)
 - insertEdge, [8](#)
 - insertVertex, [9](#)
 - isEmpty, [9](#)
 - isFull, [9](#)
 - MAX_GRAPH_SIZE, [11](#)
 - operator=, [9](#)
 - removeEdge, [10](#)
 - removeVertex, [10](#)
 - retrieveVertex, [10](#)
 - showShortestPaths, [10](#)
 - showStructure, [11](#)
 - WeightedGraph, [6, 7](#)
- WeightedGraph.cpp, [14](#)
- WeightedGraph.h, [14](#)
- WeightedGraph::Vertex, [5](#)
 - getColor, [5](#)
 - getLabel, [5](#)
 - setColor, [5](#)
 - setLabel, [5](#)