PA10

Generated by Doxygen 1.8.11

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

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

int	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

const	WeightedGraph& other

Returns

none

3.2.1.3 WeightedGraph::∼WeightedGraph ( )

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

Parameters

none	

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

none	

Returns

true or false

3.2.2.2 void WeightedGraph::clear ( )

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

Parameters

none	

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

const	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

none	

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

const	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

newVertex	

Returns

none

3.2.2.7 bool WeightedGraph::isEmpty ( ) const

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

Parameters

none	

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

none	

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

const	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

const	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

const	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

const	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

none	

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