Graf nieskierowany

Generated by Doxygen 1.8.7

Mon May 2 2016 12:43:22

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

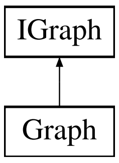
Class Documentation

4.1 Graph Class Reference

Klasa grafu.

`#include <graph.hh>`

Inheritance diagram for Graph:



Public Member Functions

- **Graph** (int vertices)
- ∼**Graph** ()
- void **add_vertex** (const int &v)
- void **add_edge** (const int &x, const int &y)
- void **remove_vertex** (const int &v)
- void **remove_edge** (const int &x, const int &y)
- **List**< int > **get_neighbours** (const int &v)
- bool **is_connected** (const int &x, const int &y)
- void **search_path_BFS** (const int &v)
- void **search_path_DFS** (const int &v)
- void **visit_DFS** (int i, const int &v)

Private Attributes

- **List**< int > ∗ **neighbours_list** =NULL
- int **size_of_neighbours_list** = 0
- int ∗ **visited** =NULL

4.1.1 Detailed Description

Klasa grafu.

Zawiera metody umożliwiające operacje na grafie.

Definition at line 12 of file graph.hh.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 Graph::Graph (int *vertices*)

Definition at line 3 of file graph.cpp.

4.1.2.2 Graph::∼Graph ( )

Definition at line 10 of file graph.cpp.

4.1.3 Member Function Documentation

4.1.3.1 void Graph::add_edge (const int & *x*, const int & *y*) `[virtual]`

Metoda dodająca krawędź do grafu.

Parameters

`in`	*element*	typu int
`in`	*element*	typu int

Implements **IGraph** (p. 12).

Definition at line 39 of file graph.cpp.

4.1.3.2 void Graph::add_vertex (const int & *v*) `[virtual]`

Metoda dodająca wierzchołek do grafu. Ma zastosowanie w przypadku dodania dodatkowego wierzchołka po procedurze inicjacji całej struktury.

Parameters

`in`	*element*	typu int

Implements **IGraph** (p. 12).

Definition at line 16 of file graph.cpp.

4.1.3.3 List< int > Graph::get_neighbours (const int & *v*) `[virtual]`

Metoda zwracająca listę sąsiadów danego wierzchołka.

Parameters

`in`	*element*	typu int

Returns

lista sąsiedztwa danego wierzchołka

Implements **IGraph** (p. 12).

Definition at line 58 of file graph.cpp.

4.1.3.4 bool Graph::is_connected (const int & *x*, const int & *y*) [virtual]

Metoda sprawdzająca istnienie krawędzi pomiędzy dwoma wierzchołkami.

Parameters

`in`	*element*	typu int
`in`	*element*	typu int

Returns

prawda lub fałsz

Implements **IGraph** (p. 12).

Definition at line 62 of file graph.cpp.

4.1.3.5 void Graph::remove_edge (const int & *x,* const int & *y*) `[virtual]`

Metoda usuwająca krawędź z grafu.

Parameters

`in`	*element*	typu int
`in`	*element*	typu int

Implements **IGraph** (p. 12).

Definition at line 52 of file graph.cpp.

4.1.3.6 void Graph::remove_vertex (const int & *v*) `[virtual]`

Metoda usuwająca wierzchołek z grafu.

Parameters

`in`	*element*	typu int

Implements **IGraph** (p. 12).

Definition at line 46 of file graph.cpp.

4.1.3.7 void Graph::search_path_BFS (const int & *v*)

Metoda przeszukująca graf wszerz

Parameters

`in`	*element*	typu int

Definition at line 71 of file graph.cpp.

4.1.3.8 void Graph::search_path_DFS (const int & *v*)

Metoda przeszukująca graf wgłąb

Parameters

`in`	*element*	typu int

Definition at line 118 of file graph.cpp.

4.1.3.9 void Graph::visit_DFS (int *i,* const int & *v*)

Metoda pomocnicza dla search_path_DFS

Parameters

in	*element*	typu int

Definition at line 104 of file graph.cpp.

4.1.4 Member Data Documentation

4.1.4.1 List<int>* Graph::neighbours_list =NULL [private]

Definition at line 15 of file graph.hh.

4.1.4.2 int Graph::size_of_neighbours_list = 0 [private]

Definition at line 16 of file graph.hh.

4.1.4.3 int* Graph::visited =NULL [private]

Definition at line 17 of file graph.hh.

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

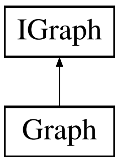
- **graph.hh**
- **graph.cpp**

4.2 IGraph Class Reference

Interfejs grafu.

`#include <igraph.hh>`

Inheritance diagram for IGraph:



Public Member Functions

- virtual void **add_vertex** (const int &v)=0
- virtual void **add_edge** (const int &x, const int &y)=0
- virtual void **remove_vertex** (const int &v)=0
- virtual void **remove_edge** (const int &x, const int &y)=0
- virtual **List**< int > **get_neighbours** (const int &v)=0
- virtual bool **is_connected** (const int &x, const int &y)=0
- virtual ~**IGraph** ()

4.2.1 Detailed Description

Interfejs grafu.

Zawiera metody umożliwiające operacje na grafie.

Definition at line 10 of file igraph.hh.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 virtual IGraph::∼IGraph ( ) `[inline]`, `[virtual]`

Definition at line 20 of file igraph.hh.

4.2.3 Member Function Documentation

4.2.3.1 virtual void IGraph::add_edge (const int & *x*, const int & *y*) `[pure virtual]`

Implemented in **Graph** (p. 8).

4.2.3.2 virtual void IGraph::add_vertex (const int & *v*) `[pure virtual]`

Implemented in **Graph** (p. 8).

4.2.3.3 virtual List<int> IGraph::get_neighbours (const int & *v*) `[pure virtual]`

Implemented in **Graph** (p. 8).

4.2.3.4 virtual bool IGraph::is_connected (const int & *x*, const int & *y*) `[pure virtual]`

Implemented in **Graph** (p. 9).

4.2.3.5 virtual void IGraph::remove_edge (const int & *x*, const int & *y*) `[pure virtual]`

Implemented in **Graph** (p. 10).

4.2.3.6 virtual void IGraph::remove_vertex (const int & *v*) `[pure virtual]`

Implemented in **Graph** (p. 10).

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

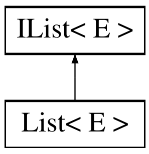
- **igraph.hh**

4.3 IList< E > Class Template Reference

Interfejs listy.

`#include <ilist.hh>`

Inheritance diagram for IList< E >:



Public Member Functions

- virtual void **add** (const E &elem, int i)=0
- virtual E **remove** (int i)=0
- virtual E **at** (int i)=0
- virtual int **size** ()=0
- virtual ~**IList** ()

4.3.1 Detailed Description

template<typename E>class IList< E >

Interfejs listy.

Zawiera metody umożliwiające operacje na liście.

Definition at line 10 of file ilist.hh.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 template<typename E> virtual IList< E >::~IList ( ) `[inline]`, `[virtual]`

Definition at line 18 of file ilist.hh.

4.3.3 Member Function Documentation

4.3.3.1 template<typename E> virtual void IList< E >::add (const E & *elem*, int *i*) `[pure virtual]`

Implemented in **List< E >** (p. 16), and **List< int >** (p. 16).

4.3.3.2 template<typename E> virtual E IList< E >::at (int *i*) `[pure virtual]`

Implemented in **List< E >** (p. 16), and **List< int >** (p. 16).

4.3.3.3 template<typename E> virtual E IList< E >::remove (int *i*) `[pure virtual]`

Implemented in **List< E >** (p. 17), and **List< int >** (p. 17).

4.3.3.4 template<typename E> virtual int IList< E >::size ( ) `[pure virtual]`

Implemented in **List< E >** (p. 17), and **List< int >** (p. 17).

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

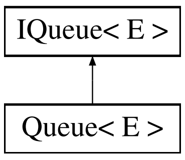
- **ilist.hh**

4.4 IQueue< E > Class Template Reference

Interfejs kolejki.

`#include <iqueue.hh>`

Inheritance diagram for IQueue< E >:



Public Member Functions

- virtual void **add** (const E &elem)=0
- virtual E **remove** ()=0
- virtual int **size** ()=0
- virtual ∼**IQueue** ()

4.4.1 Detailed Description

template<typename E>class IQueue< E >

Interfejs kolejki.

Zawiera metody umożliwiające operacje na kolejce.

Definition at line 10 of file iqueue.hh.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 template<typename E > virtual IQueue< E >::∼IQueue ( ) `[inline], [virtual]`

Definition at line 17 of file iqueue.hh.

4.4.3 Member Function Documentation

4.4.3.1 template<typename E > virtual void IQueue< E >::add (const E & *elem*) `[pure virtual]`

Implemented in **Queue< E >** (p. 21).

4.4.3.2 template<typename E > virtual E IQueue< E >::remove ( ) `[pure virtual]`

Implemented in **Queue< E >** (p. 22).

4.4.3.3 template<typename E > virtual int IQueue< E >::size ( ) `[pure virtual]`

Implemented in **Queue< E >** (p. 22).

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

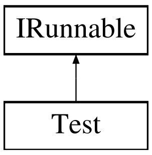
- **iqueue.hh**

4.5 IRunnable Class Reference

Interfejs klasy rozruchowej.

`#include <irunnable.hh>`

Inheritance diagram for IRunnable:



Public Member Functions

- virtual void **run** (int Argc, char ∗Argv[ ])=0

4.5.1 Detailed Description

Interfejs klasy rozruchowej.

Zawiera metodę umożliwiającą uruchomienie programu.

Definition at line 9 of file irunnable.hh.

4.5.2 Member Function Documentation

4.5.2.1 virtual void IRunnable::run (int *Argc,* char ∗ *Argv[ ]*) `[pure virtual]`

Implemented in **Test** (p. 23).

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

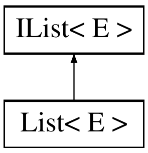
- **irunnable.hh**

4.6 List< E > Class Template Reference

Klasa listy.

`#include <list.hh>`

Inheritance diagram for List< E >:



Public Member Functions

- **List** ()
- ∼**List** ()
- void **add** (const E &elem, int i)
- E **remove** (int i)
- E **at** (int i)
- int **size** ()
- void **show_list** ()

Private Attributes

- **Node**< E > ∗ **front**
- **Node**< E > ∗ **end**
- int **list_size** =0

4.6.1 Detailed Description

template<typename E>class List< E >

Klasa listy.

Zawiera metody umożliwiające operacje na liście.

Definition at line 11 of file list.hh.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 template<typename E> List< E >::List ( ) [inline]

Definition at line 44 of file list.hh.

4.6.2.2 template<typename E> List< E >::∼List ( ) [inline]

Definition at line 48 of file list.hh.

4.6.3 Member Function Documentation

4.6.3.1 template<typename E> void List< E >::add (const E & *elem*, int *i*) [virtual]

Funkcja dodająca element do listy

Parameters

in	*element*	typu E
in	*pozycja*	i

Implements **IList**< **E** > (p. 13).

Definition at line 98 of file list.hh.

4.6.3.2 template<typename E > int List< E >::at (int *i*) [virtual]

Funkcja zwracająca element listy na danej pozycji.

Parameters

in	*pozycja*	elementu

Returns

Element typu E

Implements **IList**< **E** > (p. 13).

Definition at line 201 of file list.hh.

4.6.3.3 template<typename E > int List< E >::remove (int *i*) [virtual]

Funkcja usuwająca element z listy Wyrzuca wyjątek EmptyListException jeśli lista jest pusta oraz WrongIndex↩Exception jeśli wybrano zły indeks.

Returns

Element typu E

Implements **IList< E >** (p. 13).

Definition at line 139 of file list.hh.

4.6.3.4 template<typename E > void List< E >::show_list ( )

Funkcja wyświetlająca listę

Definition at line 212 of file list.hh.

4.6.3.5 template<typename E > int List< E >::size ( ) [virtual]

Funkcja zwracająca rozmiar listy

Returns

Rozmiar kolejki typu int

Implements **IList< E >** (p. 13).

Definition at line 196 of file list.hh.

4.6.4 Member Data Documentation

4.6.4.1 template<typename E> Node<E>* List< E >::end [private]

Wskaźnik na koniec listy

Definition at line 40 of file list.hh.

4.6.4.2 template<typename E> Node<E>* List< E >::front [private]

Wskaźnik na początek listy

Definition at line 39 of file list.hh.

4.6.4.3 template<typename E> int List< E >::list_size =0 [private]

Rozmiar listy

Definition at line 41 of file list.hh.

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

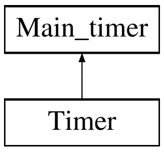
- **list.hh**
- **list.cpp**

4.7 Main_timer Class Reference

Interfejs stopera.

`#include <maintimer.hh>`

Inheritance diagram for Main_timer:



Public Member Functions

- virtual long double **get_ms_time** ()=0
- virtual void **tim_start** ()=0
- virtual void **tim_stop** ()=0
- virtual long double **return_time** ()=0
- virtual ∼**Main_timer** ()

4.7.1 Detailed Description

Interfejs stopera.

Zawiera metody umożliwiające mierzenie czasu.

Definition at line 9 of file maintimer.hh.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 virtual Main_timer::∼Main_timer ( ) `[inline], [virtual]`

Definition at line 16 of file maintimer.hh.

4.7.3 Member Function Documentation

4.7.3.1 virtual long double Main_timer::get_ms_time ( ) `[pure virtual]`

Implemented in **Timer** (p. 24).

4.7.3.2 virtual long double Main_timer::return_time ( ) `[pure virtual]`

Implemented in **Timer** (p. 24).

4.7.3.3 virtual void Main_timer::tim_start ( ) `[pure virtual]`

Implemented in **Timer** (p. 24).

4.7.3.4 virtual void Main_timer::tim_stop ( ) `[pure virtual]`

Implemented in **Timer** (p. 24).

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

- **maintimer.hh**

4.8 Node$<$ E $>$ Class Template Reference

Klasa węzła listy.

`#include <list.hh>`

Private Attributes

- E **elem**
- **Node**$<$ E $>$ $*$ **next**

Friends

- class **List**$<$ **E** $>$

4.8.1 Detailed Description

template$<$typename E$>$class Node$<$ E $>$

Klasa węzła listy.

Zawiera element węzła oraz wskaźnik na następny węzeł.

Definition at line 9 of file list.hh.

4.8.2 Friends And Related Function Documentation

4.8.2.1 template$<$typename E$>$ friend class List$<$ E $>$ `[friend]`

Definition at line 21 of file list.hh.

4.8.3 Member Data Documentation

4.8.3.1 template$<$typename E$>$ E Node$<$ E $>$::elem `[private]`

Element listy

Definition at line 24 of file list.hh.

4.8.3.2 template$<$typename E$>$ Node$<$E$>$$*$ Node$<$ E $>$::next `[private]`

Wskaźnik na kolejny węzeł

Definition at line 25 of file list.hh.

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

- **list.hh**

4.9 QNode< E > Class Template Reference

Klasa węzła kolejki.

`#include <queue.hh>`

Private Attributes

- E **elem**
- **QNode**< E > ∗ **next**

Friends

- class **Queue**< **E** >

4.9.1 Detailed Description

template<typename E>class QNode< E >

Klasa węzła kolejki.

Zawiera element węzła oraz wskaźnik na następny węzeł.

Definition at line 6 of file queue.hh.

4.9.2 Friends And Related Function Documentation

4.9.2.1 template<typename E> friend class Queue< E > `[friend]`

Definition at line 18 of file queue.hh.

4.9.3 Member Data Documentation

4.9.3.1 template<typename E> E QNode< E >::elem `[private]`

Element kolejki

Definition at line 21 of file queue.hh.

4.9.3.2 template<typename E> QNode<E>∗ QNode< E >::next `[private]`

Wskaźnik na kolejny węzeł

Definition at line 22 of file queue.hh.

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

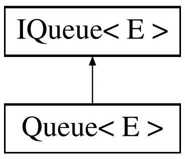
- **queue.hh**

4.10 Queue< E > Class Template Reference

Klasa kolejki.

`#include <queue.hh>`

Inheritance diagram for Queue< E >:



Public Member Functions

- **Queue** ()
- ~**Queue** ()
- void **add** (const E &elem)
- E **remove** ()
- int **size** ()
- void **show_queue** ()

Private Attributes

- **QNode**< E > ∗ **front**
- **QNode**< E > ∗ **end**
- int **queue_size** =0

4.10.1 Detailed Description

template<typename E>class Queue< E >

Klasa kolejki.

Zawiera metody umożliwiające operacje na kolejce.

Definition at line 8 of file queue.hh.

4.10.2 Constructor & Destructor Documentation

4.10.2.1 template<typename E> Queue< E >::Queue ( ) `[inline]`

Definition at line 40 of file queue.hh.

4.10.2.2 template<typename E> Queue< E >::~Queue ( ) `[inline]`

Definition at line 44 of file queue.hh.

4.10.3 Member Function Documentation

4.10.3.1 template<typename E > void Queue< E >::add (const E & *elem*) `[virtual]`

Funkcja dodająca element do kolejki

Parameters

in	*element*	typu E

Implements **IQueue**< **E** > (p. 14).

Definition at line 81 of file queue.hh.

4.10.3.2 template<typename E > E Queue< E >::remove ( ) [virtual]

Funkcja usuwająca element z kolejki Wyrzuca wyjątek EmptyQueueException jeśli kolejka jest pusta.

Returns

Element typu E

Implements **IQueue**< **E** > (p. 14).

Definition at line 99 of file queue.hh.

4.10.3.3 template<typename E > void Queue< E >::show_queue ( )

Funkcja wyświetlająca kolejkę

Definition at line 120 of file queue.hh.

4.10.3.4 template<typename E > int Queue< E >::size ( ) [virtual]

Funkcja zwracająca rozmiar kolejki

Returns

Rozmiar kolejki typu int

Implements **IQueue**< **E** > (p. 14).

Definition at line 115 of file queue.hh.

4.10.4 Member Data Documentation

4.10.4.1 template<typename E> QNode<E>* Queue< E >::end [private]

Wskaźnik na koniec kolejki

Definition at line 36 of file queue.hh.

4.10.4.2 template<typename E> QNode<E>* Queue< E >::front [private]

Wskaźnik na początek kolejki

Definition at line 35 of file queue.hh.

4.10.4.3 template<typename E> int Queue< E >::queue_size =0 [private]

Rozmiar kolejki

Definition at line 37 of file queue.hh.

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

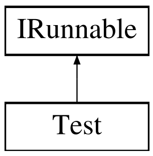
- **queue.hh**

4.11 Test Class Reference

Klasa rozruchowa.

`#include <test.hh>`

Inheritance diagram for Test:



Public Member Functions

- void **run** (int Argc, char ∗Argv[])

4.11.1 Detailed Description

Klasa rozruchowa.

Zawiera metodę umożliwiającą uruchomienie programu.

Definition at line 10 of file test.hh.

4.11.2 Member Function Documentation

4.11.2.1 void Test::run (int *Argc,* char ∗ *Argv[]*) `[virtual]`

Implements **IRunnable** (p. 15).

Definition at line 7 of file test.cpp.

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

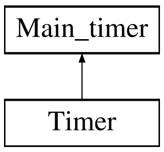
- **test.hh**
- **test.cpp**

4.12 Timer Class Reference

Klasa stopera.

`#include <timer.hh>`

Inheritance diagram for Timer:



Public Member Functions

- long double **get_ms_time** ()

- void **tim_start** ()
- void **tim_stop** ()
- long double **return_time** ()
- ∼**Timer** ()

Private Attributes

- long double **time_of_start**
- long double **time_of_stop**

4.12.1 Detailed Description

Klasa stopera.

Zawiera metody umożliwiające mierzenie czasu. Dokładny opis metod w dokumentacji projektu Lab2.

Definition at line 12 of file timer.hh.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 Timer::∼Timer ( ) `[inline]`

Definition at line 22 of file timer.hh.

4.12.3 Member Function Documentation

4.12.3.1 long double Timer::get_ms_time ( ) `[virtual]`

Implements **Main_timer** (p. 18).

Definition at line 26 of file timer.hh.

4.12.3.2 long double Timer::return_time ( ) `[virtual]`

Implements **Main_timer** (p. 18).

Definition at line 48 of file timer.hh.

4.12.3.3 void Timer::tim_start ( ) `[virtual]`

Implements **Main_timer** (p. 18).

Definition at line 36 of file timer.hh.

4.12.3.4 void Timer::tim_stop ( ) `[virtual]`

Implements **Main_timer** (p. 19).

Definition at line 42 of file timer.hh.

4.12.4 Member Data Documentation

4.12.4.1 long double Timer::time_of_start `[private]`

Definition at line 14 of file timer.hh.

4.12.4.2 long double Timer::time_of_stop [private]

Definition at line 15 of file timer.hh.

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

- **timer.hh**

Chapter 5

File Documentation

5.1 graph.cpp File Reference

```
#include "graph.hh"
```

5.2 graph.hh File Reference

```
#include "igraph.hh"
#include "list.hh"
#include "queue.hh"
```

Classes

- class **Graph**

  Klasa grafu.

5.3 igraph.hh File Reference

```
#include "list.hh"
```

Classes

- class **IGraph**

  Interfejs grafu.

5.4 ilist.hh File Reference

Classes

- class **IList**$<$ **E** $>$

  Interfejs listy.

5.5 iqueue.hh File Reference

Classes

- class **IQueue**< **E** >
  Interfejs kolejki.

5.6 irunnable.hh File Reference

Classes

- class **IRunnable**
  Interfejs klasy rozruchowej.

5.7 list.cpp File Reference

```
#include "list.hh"
#include <iostream>
```

5.8 list.hh File Reference

```
#include "ilist.hh"
#include <cstddef>
#include <cstring>
#include <iostream>
```

Classes

- class **Node**< **E** >
  Klasa węzła listy.
- class **List**< **E** >
  Klasa listy.
- class **Node**< **E** >
  Klasa węzła listy.
- class **List**< **E** >
  Klasa listy.

5.9 main.cpp File Reference

```
#include "test.hh"
#include <cstdlib>
#include <ctime>
```

Functions

- int **main** (int Argc, char ∗Argv[])

5.9.1 Function Documentation

5.9.1.1 int main (int *Argc,* char $*$ *Argv[]*)

Definition at line 5 of file main.cpp.

5.10 maintimer.hh File Reference

Classes

- class **Main_timer**

  Interfejs stopera.

5.11 queue.cpp File Reference

```
#include "queue.hh"
#include <iostream>
```

Functions

- **Queue** ()

5.11.1 Function Documentation

5.11.1.1 Queue ( )

Definition at line 4 of file queue.cpp.

5.12 queue.hh File Reference

```
#include "iqueue.hh"
```

Classes

- class **QNode**< **E** >

  Klasa węzła kolejki.
- class **Queue**< **E** >

  Klasa kolejki.
- class **QNode**< **E** >

  Klasa węzła kolejki.
- class **Queue**< **E** >

  Klasa kolejki.

5.13 test.cpp File Reference

```
#include "test.hh"
#include "graph.hh"
#include "timer.hh"
#include <cstdlib>
#include <iostream>
```

5.14 test.hh File Reference

```
#include "irunnable.hh"
```

Classes

- class **Test**

  Klasa rozruchowa.

5.15 timer.hh File Reference

```
#include <sys/time.h>
#include <cstddef>
#include "maintimer.hh"
```

Classes

- class **Timer**

  Klasa stopera.

Index