

Graf nieskierowany

Generated by Doxygen 1.8.7

Mon May 2 2016 12:43:22

Contents

1	Hierarchical Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	File Index	5
3.1	File List	5
4	Class Documentation	7
4.1	Graph Class Reference	7
4.1.1	Detailed Description	7
4.1.2	Constructor & Destructor Documentation	8
4.1.2.1	Graph	8
4.1.2.2	~Graph	8
4.1.3	Member Function Documentation	8
4.1.3.1	add_edge	8
4.1.3.2	add_vertex	8
4.1.3.3	get_neighbours	8
4.1.3.4	is_connected	9
4.1.3.5	remove_edge	10
4.1.3.6	remove_vertex	10
4.1.3.7	search_path_BFS	10
4.1.3.8	search_path_DFS	10
4.1.3.9	visit_DFS	10
4.1.4	Member Data Documentation	11
4.1.4.1	neighbours_list	11
4.1.4.2	size_of_neighbours_list	11
4.1.4.3	visited	11
4.2	IGraph Class Reference	11
4.2.1	Detailed Description	11
4.2.2	Constructor & Destructor Documentation	12

4.2.2.1	~IGraph	12
4.2.3	Member Function Documentation	12
4.2.3.1	add_edge	12
4.2.3.2	add_vertex	12
4.2.3.3	get_neighbours	12
4.2.3.4	is_connected	12
4.2.3.5	remove_edge	12
4.2.3.6	remove_vertex	12
4.3	IList< E > Class Template Reference	12
4.3.1	Detailed Description	13
4.3.2	Constructor & Destructor Documentation	13
4.3.2.1	~IList	13
4.3.3	Member Function Documentation	13
4.3.3.1	add	13
4.3.3.2	at	13
4.3.3.3	remove	13
4.3.3.4	size	13
4.4	IQueue< E > Class Template Reference	13
4.4.1	Detailed Description	14
4.4.2	Constructor & Destructor Documentation	14
4.4.2.1	~IQueue	14
4.4.3	Member Function Documentation	14
4.4.3.1	add	14
4.4.3.2	remove	14
4.4.3.3	size	14
4.5	IRunnable Class Reference	14
4.5.1	Detailed Description	15
4.5.2	Member Function Documentation	15
4.5.2.1	run	15
4.6	List< E > Class Template Reference	15
4.6.1	Detailed Description	16
4.6.2	Constructor & Destructor Documentation	16
4.6.2.1	List	16
4.6.2.2	~List	16
4.6.3	Member Function Documentation	16
4.6.3.1	add	16
4.6.3.2	at	16
4.6.3.3	remove	17
4.6.3.4	show_list	17
4.6.3.5	size	17

4.6.4	Member Data Documentation	17
4.6.4.1	end	17
4.6.4.2	front	17
4.6.4.3	list_size	17
4.7	Main_timer Class Reference	18
4.7.1	Detailed Description	18
4.7.2	Constructor & Destructor Documentation	18
4.7.2.1	~Main_timer	18
4.7.3	Member Function Documentation	18
4.7.3.1	get_ms_time	18
4.7.3.2	return_time	18
4.7.3.3	tim_start	18
4.7.3.4	tim_stop	19
4.8	Node< E > Class Template Reference	19
4.8.1	Detailed Description	19
4.8.2	Friends And Related Function Documentation	19
4.8.2.1	List< E >	19
4.8.3	Member Data Documentation	19
4.8.3.1	elem	19
4.8.3.2	next	19
4.9	QNode< E > Class Template Reference	20
4.9.1	Detailed Description	20
4.9.2	Friends And Related Function Documentation	20
4.9.2.1	Queue< E >	20
4.9.3	Member Data Documentation	20
4.9.3.1	elem	20
4.9.3.2	next	20
4.10	Queue< E > Class Template Reference	20
4.10.1	Detailed Description	21
4.10.2	Constructor & Destructor Documentation	21
4.10.2.1	Queue	21
4.10.2.2	~Queue	21
4.10.3	Member Function Documentation	21
4.10.3.1	add	21
4.10.3.2	remove	22
4.10.3.3	show_queue	22
4.10.3.4	size	22
4.10.4	Member Data Documentation	22
4.10.4.1	end	22
4.10.4.2	front	22

4.10.4.3	queue_size	22
4.11	Test Class Reference	23
4.11.1	Detailed Description	23
4.11.2	Member Function Documentation	23
4.11.2.1	run	23
4.12	Timer Class Reference	23
4.12.1	Detailed Description	24
4.12.2	Constructor & Destructor Documentation	24
4.12.2.1	~Timer	24
4.12.3	Member Function Documentation	24
4.12.3.1	get_ms_time	24
4.12.3.2	return_time	24
4.12.3.3	tim_start	24
4.12.3.4	tim_stop	24
4.12.4	Member Data Documentation	24
4.12.4.1	time_of_start	24
4.12.4.2	time_of_stop	25
5	File Documentation	27
5.1	graph.cpp File Reference	27
5.2	graph.hh File Reference	27
5.3	igraph.hh File Reference	27
5.4	ilist.hh File Reference	27
5.5	iqueue.hh File Reference	28
5.6	irunnable.hh File Reference	28
5.7	list.cpp File Reference	28
5.8	list.hh File Reference	28
5.9	main.cpp File Reference	28
5.9.1	Function Documentation	29
5.9.1.1	main	29
5.10	maintimer.hh File Reference	29
5.11	queue.cpp File Reference	29
5.11.1	Function Documentation	29
5.11.1.1	Queue	29
5.12	queue.hh File Reference	29
5.13	test.cpp File Reference	30
5.14	test.hh File Reference	30
5.15	timer.hh File Reference	30
	Index	31

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

IGraph	11
Graph	7
ICollection< E >	12
List< E >	15
ICollection< int >	12
List< int >	15
IQueue< E >	13
Queue< E >	20
IRunnable	14
Test	23
Main_timer	18
Timer	23
Node< E >	19
Node< int >	19
QNode< E >	20

Chapter 2

Class Index

2.1 Class List

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

Graph	
Klasa grafu	7
IGraph	
Interfejs grafu	11
IList< E >	
Interfejs listy	12
IQueue< E >	
Interfejs kolejki	13
IRunnable	
Interfejs klasy rozruchowej	14
List< E >	
Klasa listy	15
Main_timer	
Interfejs stopera	18
Node< E >	
Klasa węzła listy	19
QNode< E >	
Klasa węzła kolejki	20
Queue< E >	
Klasa kolejki	20
Test	
Klasa rozruchowa	23
Timer	
Klasa stopera	23

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

graph.cpp	27
graph.hh	27
igraph.hh	27
ilist.hh	27
iqueue.hh	28
irunnable.hh	28
list.cpp	28
list.hh	28
main.cpp	28
maintimer.hh	29
queue.cpp	29
queue.hh	29
test.cpp	30
test.hh	30
timer.hh	30

Chapter 4

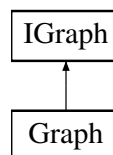
Class Documentation

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	<i>element</i>	typu int
in	<i>element</i>	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	<i>element</i>	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	<i>element</i>	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

<i>in</i>	<i>element</i>	typu int
<i>in</i>	<i>element</i>	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

<i>in</i>	<i>element</i>	typu int
<i>in</i>	<i>element</i>	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

<i>in</i>	<i>element</i>	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 wszcz

Parameters

<i>in</i>	<i>element</i>	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 wglęb

Parameters

<i>in</i>	<i>element</i>	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

<i>in</i>	<i>element</i>	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:

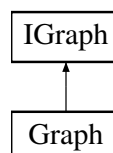
- **graph.hh**
- **graph.cpp**

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:

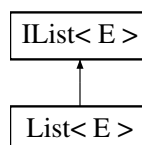
- **igraph.hh**

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:

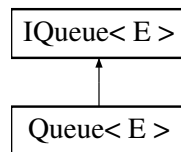
- **ilist.hh**

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:

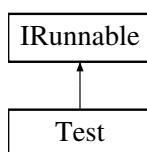
- `iqueue.hh`

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:

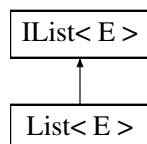
- **irunnable.hh**

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	<i>element</i>	typu E
in	<i>pozycja</i>	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	<i>pozycja</i>	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 `WrongIndexException` 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:

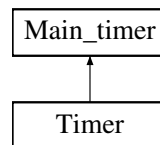
- `list.hh`
- `list.cpp`

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:

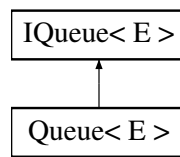
- **queue.hh**

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

<i>in</i>	<i>element</i>	typu E
-----------	----------------	--------

Implements **Queue< 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 **Queue< 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 **Queue< 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:

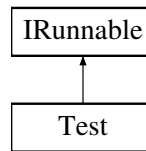
- **queue.hh**

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:

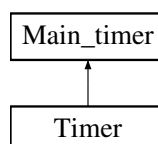
- **test.hh**
- **test.cpp**

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

- ~Graph
 - Graph, 8
- ~IGraph
 - IGraph, 12
- ~IList
 - IList, 13
- ~IQueue
 - IQueue, 14
- ~List
 - List, 16
- ~Main_timer
 - Main_timer, 18
- ~Queue
 - Queue, 21
- ~Timer
 - Timer, 24
- add
 - IList, 13
 - IQueue, 14
 - List, 16
 - Queue, 21
- add_edge
 - Graph, 8
 - IGraph, 12
- add_vertex
 - Graph, 8
 - IGraph, 12
- at
 - IList, 13
 - List, 16
- elem
 - Node, 19
 - QNode, 20
- end
 - List, 17
 - Queue, 22
- front
 - List, 17
 - Queue, 22
- get_ms_time
 - Main_timer, 18
 - Timer, 24
- get_neighbours
 - Graph, 8
 - IGraph, 12
- Graph, 7
- ~Graph, 8
- add_edge, 8
- add_vertex, 8
- get_neighbours, 8
- Graph, 8
- is_connected, 8
- neighbours_list, 11
- remove_edge, 10
- remove_vertex, 10
- search_path_BFS, 10
- search_path_DFS, 10
- size_of_neighbours_list, 11
- visit_DFS, 10
- visited, 11
- graph.cpp, 27
- graph.hh, 27
- IGraph, 11
 - ~IGraph, 12
 - add_edge, 12
 - add_vertex, 12
 - get_neighbours, 12
 - is_connected, 12
 - remove_edge, 12
 - remove_vertex, 12
- IList
 - ~IList, 13
 - add, 13
 - at, 13
 - remove, 13
 - size, 13
- IList< E >, 12
- IQueue
 - ~IQueue, 14
 - add, 14
 - remove, 14
 - size, 14
- IQueue< E >, 13
- IRunnable, 14
 - run, 15
- igraph.hh, 27
- ilist.hh, 27
- iqueue.hh, 28
- irunnable.hh, 28
- is_connected
 - Graph, 8
 - IGraph, 12
- List
 - ~List, 16

- add, 16
- at, 16
- end, 17
- front, 17
- List, 16
- list_size, 17
- remove, 16
- show_list, 17
- size, 17
- List< E >, 15
 - Node, 19
- list.cpp, 28
- list.hh, 28
- list_size
 - List, 17
- main
 - main.cpp, 29
- main.cpp, 28
 - main, 29
- Main_timer, 18
 - ~Main_timer, 18
 - get_ms_time, 18
 - return_time, 18
 - tim_start, 18
 - tim_stop, 18
- maintimer.hh, 29
- neighbours_list
 - Graph, 11
- next
 - Node, 19
 - QNode, 20
- Node
 - elem, 19
 - List< E >, 19
 - next, 19
- Node< E >, 19
- QNode
 - elem, 20
 - next, 20
 - Queue< E >, 20
- QNode< E >, 20
- Queue
 - ~Queue, 21
 - add, 21
 - end, 22
 - front, 22
 - Queue, 21
 - queue.cpp, 29
 - queue_size, 22
 - remove, 22
 - show_queue, 22
 - size, 22
- Queue< E >, 20
 - QNode, 20
- queue.cpp, 29
 - Queue, 29
- queue.hh, 29
- queue_size
 - Queue, 22
- remove
 - IList, 13
 - IQueue, 14
 - List, 16
 - Queue, 22
- remove_edge
 - Graph, 10
 - IGraph, 12
- remove_vertex
 - Graph, 10
 - IGraph, 12
- return_time
 - Main_timer, 18
 - Timer, 24
- run
 - IRunnable, 15
 - Test, 23
- search_path_BFS
 - Graph, 10
- search_path_DFS
 - Graph, 10
- show_list
 - List, 17
- show_queue
 - Queue, 22
- size
 - IList, 13
 - IQueue, 14
 - List, 17
 - Queue, 22
- size_of_neighbours_list
 - Graph, 11
- Test, 23
 - run, 23
- test.cpp, 30
- test.hh, 30
- tim_start
 - Main_timer, 18
 - Timer, 24
- tim_stop
 - Main_timer, 18
 - Timer, 24
- time_of_start
 - Timer, 24
- time_of_stop
 - Timer, 24
- Timer, 23
 - ~Timer, 24
 - get_ms_time, 24
 - return_time, 24
 - tim_start, 24
 - tim_stop, 24
 - time_of_start, 24

time_of_stop, 24
timer.hh, 30

visit_DFS
 Graph, 10
visited
 Graph, 11