

LAB2

0.1

Generated by Doxygen 1.8.6

Thu Mar 19 2015 01:29:59

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	Kolejka< T > Class Template Reference	5
3.1.1	Detailed Description	5
3.1.2	Constructor & Destructor Documentation	5
3.1.2.1	Kolejka	5
3.1.2.2	~Kolejka	5
3.1.3	Member Function Documentation	6
3.1.3.1	pop_back	6
3.1.3.2	push_front	6
3.1.3.3	show	6
3.1.3.4	size	6
3.2	List< T > Class Template Reference	6
3.2.1	Detailed Description	7
3.2.2	Constructor & Destructor Documentation	7
3.2.2.1	List	7
3.2.2.2	~List	7
3.2.3	Member Function Documentation	7
3.2.3.1	pop_back	7
3.2.3.2	pop_front	7
3.2.3.3	push_back	8
3.2.3.4	push_front	8
3.2.3.5	show	8
3.2.3.6	size	8
3.3	Node< T > Struct Template Reference	8
3.3.1	Detailed Description	9
3.3.2	Member Data Documentation	9

3.3.2.1	next	9
3.3.2.2	val	9
3.4	NodeL< T > Struct Template Reference	9
3.4.1	Detailed Description	9
3.4.2	Member Data Documentation	9
3.4.2.1	next	9
3.4.2.2	val	9
3.5	Stack< T > Class Template Reference	10
3.5.1	Detailed Description	10
3.5.2	Constructor & Destructor Documentation	10
3.5.2.1	Stack	10
3.5.2.2	~Stack	10
3.5.3	Member Function Documentation	10
3.5.3.1	peek	10
3.5.3.2	pop	11
3.5.3.3	push	11
3.5.3.4	size	11
3.5.4	Member Data Documentation	11
3.5.4.1	capacity	11
3.5.4.2	storage	11
3.5.4.3	top	11
4	File Documentation	13
4.1	inc/Kolejka.hh File Reference	13
4.1.1	Detailed Description	13
4.2	inc/Lista.hh File Reference	13
4.2.1	Detailed Description	14
4.3	inc/Stack.hh File Reference	14
4.3.1	Detailed Description	15
4.3.2	Function Documentation	15
4.3.2.1	operator<<	15
4.4	src/main.cpp File Reference	15
4.4.1	Function Documentation	16
4.4.1.1	main	16
Index		17

Chapter 1

Class Index

1.1 Class List

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

Kolejka< T >	
Klasa Kolejka (p. 5) sluzi do wykonywania podstawowych operacji na Kolejce: dodaj,odejmij element. Przechowuje informacje o ilosci wszystkich elementow	5
List< T >	
Klasa List (p. 6) sluzi do wykonywania podstawowych operacji na Liscie: dodaj,odejmij element. Przechowuje informacje o ilosci wszystkich elementow	6
Node< T >	8
NodeL< T >	9
Stack< T >	10

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

inc/ Kolejka.hh	
Struktura przechowujaca wartosc wezla i wskaznik na nastepny element typu Node (p. 8)	13
inc/ Lista.hh	
Struktura przechowujaca wartosc wezla i wskaznik na nastepny element typu Node (p. 8)	13
inc/ Stack.hh	
Klasa Stack (p. 10) sluzi do przechowywania, dodawania, zdejmowania kolejnych elementow stosu	14
src/ main.cpp	15

Chapter 3

Class Documentation

3.1 Kolejka< T > Class Template Reference

klasa **Kolejka** (p. 5) sluzy do wykonywania podstawowych operacji na Kolejce: dodaj,odejmij element. Przechowuje informacje o ilosci wszystkich elementow.

```
#include <Kolejka.hh>
```

Public Member Functions

- **Kolejka** ()
- **~Kolejka** ()
- int **size** ()
- void **push_front** (T value)
- void **pop_back** ()
- void **show** ()

3.1.1 Detailed Description

```
template<typename T>class Kolejka< T >
```

klasa **Kolejka** (p. 5) sluzy do wykonywania podstawowych operacji na Kolejce: dodaj,odejmij element. Przechowuje informacje o ilosci wszystkich elementow.

Definition at line 21 of file Kolejka.hh.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 template<typename T > Kolejka< T >::Kolejka ()

brief Konstruktor bezparametryczny

Konstruktor bezparametryczny, ustawia parametry na 0

Definition at line 59 of file Kolejka.hh.

3.1.2.2 template<typename T > Kolejka< T >::~~Kolejka ()

Destruktor, usuwa kolejne elementy kolejki zaczynajac od poczatku

Definition at line 69 of file Kolejka.hh.

3.1.3 Member Function Documentation

3.1.3.1 `template<typename T> void Kolejka<T>::pop_back ()`

brief Funkcja zdejmuję element z końca kolejki

Funkcja usuwa element z końca kolejki

Precondition

Kolejka (p. 5) nie może być pusta

Definition at line 106 of file Kolejka.hh.

3.1.3.2 `template<typename T> void Kolejka<T>::push_front (T value)`

brief Funkcja dodaje element na początek kolejki

Funkcja służy do dodania elementu na początek kolejki

Parameters

<i>in</i>	<i>value-ty</i>	int, wartość elementu zmiennej dodanej do kolejki
-----------	-----------------	---

Definition at line 92 of file Kolejka.hh.

3.1.3.3 `template<typename T> void Kolejka<T>::show ()`

brief Funkcja wyświetla wszystkie elementy na standardowe wyjście

Funkcja wyświetla elementy kolejki

Definition at line 134 of file Kolejka.hh.

3.1.3.4 `template<typename T> int Kolejka<T>::size ()`

brief Funkcja zwraca rozmiar kolejki

Returns

Funkcja zwraca wartość rozmiaru kolejki

Definition at line 82 of file Kolejka.hh.

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

- `inc/Kolejka.hh`

3.2 `List<T>` Class Template Reference

klasa **List** (p. 6) służy do wykonywania podstawowych operacji na liście: dodaj, odejmij element. Przechowuje informacje o ilości wszystkich elementów.

```
#include <Lista.hh>
```

Public Member Functions

- `List ()`

- `~List ()`
- `int size ()`
- `void push_front (T value)`
- `void pop_front ()`
- `void push_back (T value)`
- `void pop_back ()`
- `void show ()`

3.2.1 Detailed Description

`template<typename T>class List< T >`

klasa **List** (p.6) sluzy do wykonywania podstawowych operacji na Liscie: dodaj,odejmij element. Przechowuje informacje o ilosci wszystkich elementow.

Definition at line 23 of file Lista.hh.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 `template<typename T > List< T >::List ()`

brief Konstruktor bezparametryczny

Konstruktor bezparametryczny, ustawia parametry na 0

Definition at line 73 of file Lista.hh.

3.2.2.2 `template<typename T > List< T >::~~List ()`

brief Destruktor

Destruktor, usuwa kolejne elementy listy zaczynajac od poczatku

Definition at line 83 of file Lista.hh.

3.2.3 Member Function Documentation

3.2.3.1 `template<typename T > void List< T >::pop_back ()`

brief Funkcja zdejmuj element z konca listy

Funkcja usuwa element z konca listy

Precondition

Lista nie moze byc pusta

Definition at line 159 of file Lista.hh.

3.2.3.2 `template<typename T > void List< T >::pop_front ()`

brief Funkcja zdejmuj element z poczatku listy

Funkcja usuwa element z poczatku listy

Precondition

Lista nie moze byc pusta

Definition at line 120 of file Lista.hh.

3.2.3.3 `template<typename T > void List< T >::push_back (T value)`

brief Funkcja dodaje element na koniec listy

Funkcja dodaje element na koniec listy

Parameters

<i>in</i>	<i>value</i>	- typ int, wartosc elementu dodanego na koniec listy
-----------	--------------	--

Definition at line 138 of file Lista.hh.

3.2.3.4 `template<typename T > void List< T >::push_front (T value)`

brief Funkcja dodaje element na poczatek listy

Funkcja sluzzy do dodania elementu na poczatek listy

Parameters

<i>in</i>	<i>value-ty</i>	int, wartosc elementu zmiennej dodanej do listy
-----------	-----------------	---

Definition at line 106 of file Lista.hh.

3.2.3.5 `template<typename T > void List< T >::show ()`

brief Funkcja wyswietla wszystkie elementy na standardowe wyjscie

Funkcja wyswietla elementy listy

Definition at line 187 of file Lista.hh.

3.2.3.6 `template<typename T > int List< T >::size ()`

brief Funkcja zwraca rozmiar listy

Returns

Funkcja zwraca wartosc rozmiaru listy

Definition at line 96 of file Lista.hh.

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

- inc/**Lista.hh**

3.3 `Node< T >` Struct Template Reference

```
#include <Kolejka.hh>
```

Public Attributes

- **T val**
- **Node< T > * next**

3.3.1 Detailed Description

`template<typename T> struct Node< T >`

Definition at line 10 of file Kolejka.hh.

3.3.2 Member Data Documentation

3.3.2.1 `template<typename T> Node<T>* Node< T >::next`

Definition at line 13 of file Kolejka.hh.

3.3.2.2 `template<typename T> T Node< T >::val`

Definition at line 12 of file Kolejka.hh.

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

- inc/**Kolejka.hh**

3.4 NodeL< T > Struct Template Reference

```
#include <Lista.hh>
```

Public Attributes

- **T val**
- **NodeL< T > * next**

3.4.1 Detailed Description

`template<typename T> struct NodeL< T >`

Definition at line 11 of file Lista.hh.

3.4.2 Member Data Documentation

3.4.2.1 `template<typename T> NodeL<T>* NodeL< T >::next`

Definition at line 14 of file Lista.hh.

3.4.2.2 `template<typename T> T NodeL< T >::val`

Definition at line 13 of file Lista.hh.

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

- inc/**Lista.hh**

3.5 Stack< T > Class Template Reference

```
#include <Stack.hh>
```

Public Member Functions

- **Stack** (int **capacity**)
- void **push** (T value)
- T **peek** ()
- int **size** ()
- ~**Stack** ()
- void **pop** ()

Public Attributes

- T * **top**
- int **capacity**
- T * **storage**

3.5.1 Detailed Description

```
template<typename T>class Stack< T >
```

Definition at line 10 of file Stack.hh.

3.5.2 Constructor & Destructor Documentation

3.5.2.1 `template<typename T > Stack< T >::Stack (int capacity)`

Konstruktor parametryczny klasy **Stack** (p. 10)

Parameters

<i>in</i>	<i>capacity</i>	- typ int, rozmiar stosu
-----------	-----------------	--------------------------

Definition at line 64 of file Stack.hh.

3.5.2.2 `template<typename T > Stack< T >::~Stack ()`

Destruktor klasy **Stack** (p. 10)

Definition at line 122 of file Stack.hh.

3.5.3 Member Function Documentation

3.5.3.1 `template<typename T > T Stack< T >::peek ()`

Funkcja pokazuje element znajdujący się na szczycie stosu

Precondition

Stos nie może być pusty

Definition at line 102 of file Stack.hh.

3.5.3.2 `template<typename T> void Stack< T >::pop ()`

Funkcja pop zdejmuję ostatni element ze stosu

Precondition

Stos nie może być pusty

Definition at line 133 of file Stack.hh.

3.5.3.3 `template<typename T> void Stack< T >::push (T value)`

Funkcja dodaje element na koniec tablicy stosu

Parameters

<i>in</i>	<i>value</i>	- typ int, wartość dodana do stosu
-----------	--------------	------------------------------------

Postcondition

wykorzystana metoda podwajania do powiększania stosu

Definition at line 81 of file Stack.hh.

3.5.3.4 `template<typename T> int Stack< T >::size ()`

Returns

zwraca ilość elementów stosu

Definition at line 113 of file Stack.hh.

3.5.4 Member Data Documentation

3.5.4.1 `template<typename T> int Stack< T >::capacity`

Definition at line 20 of file Stack.hh.

3.5.4.2 `template<typename T> T* Stack< T >::storage`

Definition at line 24 of file Stack.hh.

3.5.4.3 `template<typename T> T* Stack< T >::top`

Definition at line 16 of file Stack.hh.

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

- inc/Stack.hh

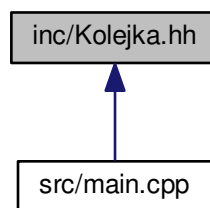
Chapter 4

File Documentation

4.1 inc/Kolejka.hh File Reference

Struktura przechowująca wartość węzła i wskaźnik na następny element typu **Node** (p. 8).

This graph shows which files directly or indirectly include this file:



Classes

- struct **Node**< T >
- class **Kolejka**< T >

*klasa **Kolejka** (p. 5) służy do wykonywania podstawowych operacji na Kolejce: dodaj, odejmij element. Przechowuje informacje o ilości wszystkich elementów.*

4.1.1 Detailed Description

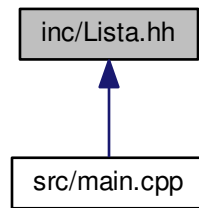
Struktura przechowująca wartość węzła i wskaźnik na następny element typu **Node** (p. 8).

Definition in file **Kolejka.hh**.

4.2 inc/Lista.hh File Reference

Struktura przechowująca wartość węzła i wskaźnik na następny element typu **Node** (p. 8).

This graph shows which files directly or indirectly include this file:



Classes

- struct **NodeL**< T >
- class **List**< T >

*klasa **List** (p. 6) służy do wykonywania podstawowych operacji na Liscie: dodaj,odejmij element. Przechowuje informacje o ilosci wszystkich elementow.*

4.2.1 Detailed Description

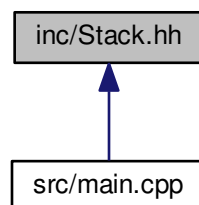
Struktura przechowujaca wartosc wezla i wskaznik na nastepny element typu **Node** (p. 8).

Definition in file **Lista.hh**.

4.3 inc/Stack.hh File Reference

Klasa **Stack** (p. 10) służy do przechowywania, dodawania,zdejmowania kolejnych elementow stosu.

This graph shows which files directly or indirectly include this file:



Classes

- class **Stack**< T >

Functions

- `template<typename T >`
`std::ostream & operator<< (std::ostream &out, const Stack< T > &stack)`

4.3.1 Detailed Description

Klasa **Stack** (p. 10) sluży do przechowywania, dodawania, zdejmowania kolejnych elementów stosu.

Definition in file **Stack.hh**.

4.3.2 Function Documentation

4.3.2.1 `template<typename T > std::ostream& operator<< (std::ostream & out, const Stack< T > & stack)`

Funkcja operatorowa służy do wyświetlania stosu, zbudna

Parameters

<code>in</code>	<code>&out</code>	- referencja do strumienia wyjściowego
<code>in</code>	<code>&stack-</code>	referencja do stosu

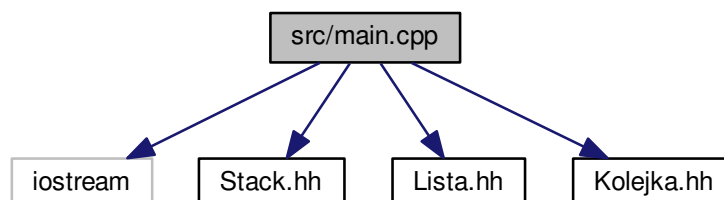
Returns

zwraca referencje do strumienia wyjściowego

Definition at line 148 of file Stack.hh.

4.4 src/main.cpp File Reference

```
#include <iostream>
#include "Stack.hh"
#include "Lista.hh"
#include "Kolejka.hh"
Include dependency graph for main.cpp:
```



Functions

- `int main ()`

4.4.1 Function Documentation

4.4.1.1 `int main ()`

Definition at line 6 of file main.cpp.

Index

- ~Kolejka
 - Kolejka, 5
- ~List
 - List, 7
- ~Stack
 - Stack, 10
- capacity
 - Stack, 11
- inc/Kolejka.hh, 13
- inc/Lista.hh, 13
- inc/Stack.hh, 14
- Kolejka
 - ~Kolejka, 5
 - Kolejka, 5
 - pop_back, 6
 - push_front, 6
 - show, 6
 - size, 6
- Kolejka< T >, 5
- List
 - ~List, 7
 - List, 7
 - pop_back, 7
 - pop_front, 7
 - push_back, 8
 - push_front, 8
 - show, 8
 - size, 8
- List< T >, 6
- main
 - main.cpp, 16
- main.cpp
 - main, 16
- next
 - Node, 9
 - NodeL, 9
- Node
 - next, 9
 - val, 9
- Node< T >, 8
- NodeL
 - next, 9
 - val, 9
- NodeL< T >, 9

- operator<<
 - Stack.hh, 15
- peek
 - Stack, 10
- pop
 - Stack, 10
- pop_back
 - Kolejka, 6
 - List, 7
- pop_front
 - List, 7
- push
 - Stack, 11
- push_back
 - List, 8
- push_front
 - Kolejka, 6
 - List, 8
- show
 - Kolejka, 6
 - List, 8
- size
 - Kolejka, 6
 - List, 8
 - Stack, 11
- src/main.cpp, 15
- Stack
 - ~Stack, 10
 - capacity, 11
 - peek, 10
 - pop, 10
 - push, 11
 - size, 11
 - Stack, 10
 - storage, 11
 - top, 11
- Stack< T >, 10
- Stack.hh
 - operator<<, 15
- storage
 - Stack, 11
- top
 - Stack, 11
- val
 - Node, 9
 - NodeL, 9