

Lista Dwukierunkowa

Generated by Doxygen 1.12.0

1 Class Index	1
1.1 Class List	1
2 File Index	3
2.1 File List	3
3 Class Documentation	5
3.1 lista_dwukierunkowa Class Reference	5
3.1.1 Detailed Description	6
3.1.2 Constructor & Destructor Documentation	6
3.1.2.1 lista_dwukierunkowa()	6
3.1.2.2 ~lista_dwukierunkowa()	6
3.1.3 Member Function Documentation	6
3.1.3.1 dodajindex()	6
3.1.3.2 dodajkoniec()	6
3.1.3.3 dodajpoczątek()	7
3.1.3.4 minusindex()	7
3.1.3.5 minuskoniec()	7
3.1.3.6 minuspoczątek()	7
3.1.3.7 usun()	7
3.1.3.8 wypisz()	8
3.1.3.9 wypisznastepny_element()	8
3.1.3.10 wypiszodtylu()	8
3.1.3.11 wypiszpoprzedni_element()	8
3.2 wezel Class Reference	8
3.2.1 Detailed Description	9
3.2.2 Constructor & Destructor Documentation	9
3.2.2.1 wezel()	9
3.2.3 Member Data Documentation	9
3.2.3.1 a	9
3.2.3.2 nastepny_element	9
3.2.3.3 poprzedni_element	9
4 File Documentation	11
4.1 Lista_dwukierunkowa_/Lista_dwukierunkowa_.cpp File Reference	11
4.1.1 Function Documentation	11
4.1.1.1 main()	11
4.2 Lista_dwukierunkowa_/metody.cpp File Reference	11
4.3 Lista_dwukierunkowa_/naglowek.h File Reference	11
4.3.1 Detailed Description	12
4.4 naglowek.h	12
Index	13

Chapter 1

Class Index

1.1 Class List

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

lista_dwukierunkowa	Klasa lista_dwukierunkowa reprezentujaca liste dwukierunkowa	5
wezel	Klasa wezel reprezentujaca wezel	8

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Lista_dwukierunkowa_/Lista_dwukierunkowa_.cpp	11
Lista_dwukierunkowa_/metody.cpp	11
Lista_dwukierunkowa_/naglowek.h	
Plik naglowkowy zawierajacy definicje klas wezel i lista_dwukierunkowa dla dwukierunkowej listy	11

Chapter 3

Class Documentation

3.1 lista_dwukierunkowa Class Reference

Klasa [lista_dwukierunkowa](#) reprezentujaca liste dwukierunkowa.

```
#include <naglowek.h>
```

Public Member Functions

- [lista_dwukierunkowa](#) (int a)
Konstruktor listy dwukierunkowej, który tworzy listę z jednym elementem.
- [~lista_dwukierunkowa](#) (void)
Destruktor listy. Usuwa wszystkie elementy listy i zwalnia pamięć.
- void [dodajpoczatek](#) (int a)
Dodaje nowy element o wartości a na początku listy.
- void [dodajkoniec](#) (int a)
Dodaje nowy element o wartości a na końcu listy.
- void [minuspoczatek](#) (void)
Usuwa element z początku listy.
- void [minuskoniec](#) (void)
Usuwa element z końca listy.
- void [dodajindex](#) (int a, int b)
Dodaje nowy element o wartości a na pozycji b.
- void [minusindex](#) (int a)
Usuwa element z podanego indeksu.
- void [wypisz](#) (void)
Wypisuje zawartość listy od początku do końca.
- void [wypiszodtylu](#) (void)
Wypisuje zawartość listy od końca do początku.
- void [wypisznastepny_element](#) (int a)
Wypisuje wartość elementu, który znajduje się po elemencie o wartości a.
- void [wypiszpoprzedni_element](#) (int a)
Wypisuje wartość elementu, który znajduje się przed elementem o wartości a.
- void [usun](#) (void)
Usuwa wszystkie elementy z listy.

3.1.1 Detailed Description

Klasa [lista_dwukierunkowa](#) reprezentujaca liste dwukierunkowa.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 lista_dwukierunkowa()

```
lista_dwukierunkowa::lista_dwukierunkowa (
    int a)
```

Konstruktor listy dwukierunkowej, ktory tworzy liste z jednym elementem.

Parameters

<i>a</i>	Wartosc pierwszego elementu listy.
----------	------------------------------------

3.1.2.2 ~lista_dwukierunkowa()

```
lista_dwukierunkowa::~~lista_dwukierunkowa (
    void )
```

Destruktor listy. Usuwa wszystkie elementy listy i zwalnia pamiec.

3.1.3 Member Function Documentation

3.1.3.1 dodajindex()

```
void lista_dwukierunkowa::dodajindex (
    int a,
    int b)
```

Dodaje nowy element o wartosci a na pozycji b.

Parameters

<i>a</i>	Wartosc nowego elementu.
<i>b</i>	Indeks, na ktorym nalezy wstawic nowy element.

3.1.3.2 dodajkoniec()

```
void lista_dwukierunkowa::dodajkoniec (
    int a)
```

Dodaje nowy element o wartosci a na koncu listy.

Parameters

<i>a</i>	Wartosc nowego elementu.
----------	--------------------------

3.1.3.3 dodajpoczatek()

```
void lista_dwukierunkowa::dodajpoczatek (
    int a)
```

Dodaje nowy element o wartosci a na poczatku listy.

Parameters

<i>a</i>	Wartosc nowego elementu.
----------	--------------------------

3.1.3.4 minusindex()

```
void lista_dwukierunkowa::minusindex (
    int a)
```

Usuwa element z podanego indeksu.

Parameters

<i>a</i>	Indeks elementu, który ma zostać usunięty.
----------	--

3.1.3.5 minuskoniec()

```
void lista_dwukierunkowa::minuskoniec (
    void )
```

Usuwa element z końca listy.

3.1.3.6 minuspoczatek()

```
void lista_dwukierunkowa::minuspoczatek (
    void )
```

Usuwa element z początku listy.

3.1.3.7 usun()

```
void lista_dwukierunkowa::usun (
    void )
```

Usuwa wszystkie elementy z listy.

3.1.3.8 wypisz()

```
void lista_dwukierunkowa::wypisz (
    void )
```

Wypisuje zawartosc listy od poczatku do konca.

3.1.3.9 wypisznastepny_element()

```
void lista_dwukierunkowa::wypisznastepny_element (
    int a)
```

Wypisuje wartosc elementu, ktory znajduje sie po elemencie o wartosci a.

Parameters

<i>a</i>	Wartosc elementu, po ktorym ma zostac wypisany nastepny element.
----------	--

3.1.3.10 wypiszodtylu()

```
void lista_dwukierunkowa::wypiszodtylu (
    void )
```

Wypisuje zawartosc listy od konca do poczatku.

3.1.3.11 wypiszpoprzedni_element()

```
void lista_dwukierunkowa::wypiszpoprzedni_element (
    int a)
```

Wypisuje wartosc elementu, ktory znajduje sie przed elementem o wartosci a.

Parameters

<i>a</i>	Wartosc elementu, przed ktorym ma zostac wypisany poprzedni element.
----------	--

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

- Lista_dwukierunkowa_/naglowek.h
- Lista_dwukierunkowa_/metody.cpp

3.2 wezel Class Reference

Klasa wezel reprezentujaca wezel.

```
#include <naglowek.h>
```

Public Member Functions

- [wezel](#) (int [a](#))
Konstruktor z parametrem.

Public Attributes

- int [a](#)
Wartosc przechowywana w wezelorce listy.
- [wezel](#) * [nastepny_element](#)
Wskaźnik na następny element listy.
- [wezel](#) * [poprzedni_element](#)
Wskaźnik na poprzedni listy.

3.2.1 Detailed Description

Klasa wezel reprezentująca wezel.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 wezel()

```
wezel::wezel (  
    int a)
```

Konstruktor z parametrem.

Parameters

a	Wartosc, która należy przypisać do pola a.
-------------------	--

3.2.3 Member Data Documentation

3.2.3.1 a

```
int wezel::a
```

Wartosc przechowywana w wezelorce listy.

3.2.3.2 nastepny_element

```
wezel* wezel::nastepny_element
```

Wskaźnik na następny element listy.

3.2.3.3 poprzedni_element

```
wezel* wezel::poprzedni_element
```

Wskaźnik na poprzedni listy.

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

- Lista_dwukierunkowa_/naglowek.h
- Lista_dwukierunkowa_/metody.cpp

Chapter 4

File Documentation

4.1 Lista_dwukierunkowa_/Lista_dwukierunkowa_.cpp File Reference

```
#include <iostream>
#include "naglowek.h"
```

Functions

- int [main](#) ()

4.1.1 Function Documentation

4.1.1.1 main()

```
int main ()
```

4.2 Lista_dwukierunkowa_/metody.cpp File Reference

```
#include "naglowek.h"
#include <iostream>
```

4.3 Lista_dwukierunkowa_/naglowek.h File Reference

Plik naglowkowy zawierajacy definicje klas wezel i [lista_dwukierunkowa](#) dla dwukierunkowej listy.

Classes

- class [wezel](#)
Klasa wezel reprezentujaca wezel.
- class [lista_dwukierunkowa](#)
Klasa lista_dwukierunkowa reprezentujaca liste dwukierunkowa.

4.3.1 Detailed Description

Plik naglowkowy zawierajacy definicje klas wezel i [lista_dwukierunkowa](#) dla dwukierunkowej listy.

4.4 naglowek.h

[Go to the documentation of this file.](#)

```
00001 #pragma once
00010 class wezel {
00011 public:
00012     int a;
00013     wezel* nastepny_element;
00014     wezel* poprzedni_element;
00019     wezel(int a);
00020 };
00024 class lista_dwukierunkowa {
00025 private:
00026     wezel* poczatek;
00027     wezel* koniec;
00028
00029 public:
00034     lista_dwukierunkowa(int a);
00038     ~lista_dwukierunkowa(void);
00043     void dodajpoczatek(int a);
00048     void dodajkoniec(int a);
00052     void minuspoczatek(void);
00056     void minuskoniec(void);
00062     void dodajindex(int a, int b);
00067     void minusindex(int a);
00071     void wypisz(void);
00075     void wypiszodtylu(void);
00080     void wypisznastepny_element(int a);
00085     void wypiszpoprzedni_element(int a);
00089     void usun(void);
00090 };
```


Index

~lista_dwukierunkowa
lista_dwukierunkowa, 6

a
wezel, 9

dodajindex
lista_dwukierunkowa, 6

dodajkoniec
lista_dwukierunkowa, 6

dodajpoczątek
lista_dwukierunkowa, 7

lista_dwukierunkowa, 5
~lista_dwukierunkowa, 6
dodajindex, 6
dodajkoniec, 6
dodajpoczątek, 7
lista_dwukierunkowa, 6
minusindex, 7
minuskoniec, 7
minuspoczątek, 7
usun, 7
wypisz, 7
wypisznastepny_element, 8
wypiszodtylu, 8
wypiszpoprzedni_element, 8

Lista_dwukierunkowa.cpp
main, 11

Lista_dwukierunkowa_/Lista_dwukierunkowa_.cpp, 11

Lista_dwukierunkowa_/metody.cpp, 11

Lista_dwukierunkowa_/naglowek.h, 11, 12

main
Lista_dwukierunkowa_.cpp, 11

minusindex
lista_dwukierunkowa, 7

minuskoniec
lista_dwukierunkowa, 7

minuspoczątek
lista_dwukierunkowa, 7

nastepny_element
wezel, 9

poprzedni_element
wezel, 9

usun
lista_dwukierunkowa, 7

wezel, 8
a, 9
nastepny_element, 9
poprzedni_element, 9
wezel, 9

wypisz
lista_dwukierunkowa, 7

wypisznastepny_element
lista_dwukierunkowa, 8

wypiszodtylu
lista_dwukierunkowa, 8

wypiszpoprzedni_element
lista_dwukierunkowa, 8