

Zadanie 8

Generated by Doxygen 1.8.9.1

Wed May 27 2015 22:46:25

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	Algorytm< T > Class Template Reference	7
4.1.1	Detailed Description	7
4.1.2	Constructor & Destructor Documentation	7
4.1.2.1	~Algorytm	7
4.1.3	Member Function Documentation	7
4.1.3.1	alokujdane	7
4.1.3.2	wykonajalgorytm	8
4.2	Algorytm1 Class Reference	8
4.2.1	Detailed Description	9
4.2.2	Constructor & Destructor Documentation	9
4.2.2.1	~Algorytm1	9
4.2.3	Member Function Documentation	9
4.2.3.1	alokujdane	9
4.2.3.2	wykonajalgorytm	9
4.3	Algorytm2 Class Reference	10
4.3.1	Detailed Description	11
4.3.2	Member Function Documentation	11
4.3.2.1	alokujdane	11
4.3.2.2	wykonajalgorytm	11
4.4	Algorytm3 Class Reference	11
4.4.1	Detailed Description	12
4.4.2	Member Function Documentation	12

4.4.2.1	alokujdane	12
4.4.2.2	wykonajalgorytm	13
4.5	Algorytm4 Class Reference	13
4.5.1	Detailed Description	14
4.5.2	Constructor & Destructor Documentation	14
4.5.2.1	~Algorytm4	14
4.5.3	Member Function Documentation	14
4.5.3.1	alokujdane	15
4.5.3.2	mergesort	16
4.5.3.3	scal	16
4.5.3.4	wykonajalgorytm	16
4.6	Algorytm5 Class Reference	17
4.6.1	Detailed Description	18
4.6.2	Member Function Documentation	18
4.6.2.1	alokujdane	18
4.6.2.2	wykonajalgorytm	19
4.7	Algorytm6 Class Reference	20
4.7.1	Detailed Description	21
4.7.2	Constructor & Destructor Documentation	21
4.7.2.1	~Algorytm6	21
4.7.3	Member Function Documentation	21
4.7.3.1	alokujdane	21
4.7.3.2	wykonajalgorytm	22
4.8	ArrayLista Class Reference	22
4.8.1	Detailed Description	24
4.8.2	Constructor & Destructor Documentation	24
4.8.2.1	ArrayLista	24
4.8.2.2	ArrayLista	24
4.8.2.3	~ArrayLista	24
4.8.3	Member Function Documentation	24
4.8.3.1	pop	24
4.8.3.2	push	24
4.8.3.3	push	24
4.8.3.4	size	24
4.9	Benchmark< T > Class Template Reference	25
4.9.1	Detailed Description	26
4.9.2	Member Function Documentation	26
4.9.2.1	powiadam	26
4.9.2.2	testuj	26
4.10	DrzewoBinarne Class Reference	27

4.10.1 Detailed Description	28
4.10.2 Constructor & Destructor Documentation	28
4.10.2.1 DrzewoBinarne	28
4.10.2.2 ~DrzewoBinarne	28
4.10.3 Member Function Documentation	28
4.10.3.1 obroc_l	28
4.10.3.2 obroc_p	29
4.10.3.3 pop	29
4.10.3.4 pop	29
4.10.3.5 push	29
4.10.3.6 rownowaz	30
4.10.3.7 size	30
4.10.3.8 wypisz	30
4.10.3.9 wypisz_pelne	31
4.11 DrzewoRB Class Reference	31
4.11.1 Detailed Description	32
4.11.2 Constructor & Destructor Documentation	32
4.11.2.1 DrzewoRB	32
4.11.2.2 ~DrzewoRB	33
4.11.3 Member Function Documentation	33
4.11.3.1 obroc_l	33
4.11.3.2 obroc_p	33
4.11.3.3 pop	33
4.11.3.4 pop	33
4.11.3.5 push	34
4.11.3.6 size	34
4.11.3.7 ukladaj	34
4.11.3.8 wypisz	35
4.11.3.9 wypisz_pelne	35
4.12 HaszTab Class Reference	35
4.12.1 Detailed Description	37
4.12.2 Constructor & Destructor Documentation	37
4.12.2.1 HaszTab	37
4.12.2.2 ~HaszTab	37
4.12.3 Member Function Documentation	37
4.12.3.1 mieszaj	37
4.12.3.2 odczytaj	37
4.12.3.3 pop	38
4.12.3.4 pop	38
4.12.3.5 push	38

4.12.3.6	push	39
4.12.3.7	size	39
4.12.3.8	size_k1	39
4.12.3.9	size_k2	39
4.13	Kolejka Class Reference	40
4.13.1	Detailed Description	41
4.13.2	Constructor & Destructor Documentation	41
4.13.2.1	Kolejka	41
4.13.2.2	~Kolejka	41
4.13.3	Member Function Documentation	41
4.13.3.1	pop	41
4.13.3.2	pop	41
4.13.3.3	push	41
4.13.3.4	size	41
4.14	Lista Class Reference	42
4.14.1	Detailed Description	43
4.14.2	Constructor & Destructor Documentation	43
4.14.2.1	Lista	43
4.14.2.2	~Lista	43
4.14.3	Member Function Documentation	43
4.14.3.1	pop	43
4.14.3.2	pop	43
4.14.3.3	push	44
4.14.3.4	push	45
4.14.3.5	size	45
4.15	Obserwator Class Reference	45
4.15.1	Detailed Description	46
4.15.2	Member Function Documentation	46
4.15.2.1	odswiez	46
4.16	ObserwatorZapisujacy Class Reference	46
4.16.1	Detailed Description	47
4.16.2	Member Function Documentation	47
4.16.2.1	odswiez	47
4.17	Obserwowany Class Reference	47
4.17.1	Detailed Description	48
4.17.2	Member Function Documentation	48
4.17.2.1	dodaj	49
4.17.2.2	usun	50
4.17.3	Member Data Documentation	50
4.17.3.1	obserwatorzy	50

4.18 Stos Class Reference	50
4.18.1 Detailed Description	51
4.18.2 Constructor & Destructor Documentation	51
4.18.2.1 Stos	51
4.18.2.2 ~Stos	51
4.18.3 Member Function Documentation	51
4.18.3.1 pop	51
4.18.3.2 pop	52
4.18.3.3 push	52
4.18.3.4 size	52
4.19 Zasobnik< T > Class Template Reference	52
4.19.1 Detailed Description	53
4.19.2 Constructor & Destructor Documentation	53
4.19.2.1 ~Zasobnik	53
4.19.3 Member Function Documentation	53
4.19.3.1 pop	53
4.19.3.2 pop	53
4.19.3.3 push	53
4.19.3.4 size	54
5 File Documentation	55
5.1 Algorytm1.cpp File Reference	55
5.1.1 Detailed Description	55
5.2 Algorytm1.hh File Reference	55
5.2.1 Detailed Description	56
5.3 Algorytm2.cpp File Reference	56
5.3.1 Detailed Description	57
5.4 Algorytm2.hh File Reference	57
5.4.1 Detailed Description	58
5.5 Algorytm3.cpp File Reference	58
5.5.1 Detailed Description	59
5.6 Algorytm3.hh File Reference	59
5.6.1 Detailed Description	60
5.7 Algorytm4.cpp File Reference	60
5.7.1 Detailed Description	61
5.8 Algorytm4.hh File Reference	61
5.8.1 Detailed Description	62
5.9 Algorytm5.cpp File Reference	62
5.9.1 Detailed Description	63
5.10 Algorytm5.hh File Reference	63

5.10.1 Detailed Description	64
5.11 Algorytm6.cpp File Reference	64
5.11.1 Detailed Description	65
5.12 Algorytm6.hh File Reference	65
5.12.1 Detailed Description	66
5.13 AlgorytmAbs.hh File Reference	66
5.13.1 Detailed Description	67
5.14 ArrayLista.cpp File Reference	67
5.14.1 Detailed Description	68
5.15 ArrayLista.hh File Reference	68
5.15.1 Detailed Description	69
5.16 Benchmark.cpp File Reference	69
5.16.1 Detailed Description	69
5.17 Benchmark.hh File Reference	70
5.17.1 Detailed Description	70
5.18 DrzewoBinarne.cpp File Reference	71
5.18.1 Detailed Description	71
5.19 DrzewoBinarne.hh File Reference	71
5.19.1 Detailed Description	72
5.20 DrzewoRB.cpp File Reference	72
5.20.1 Detailed Description	73
5.21 DrzewoRB.hh File Reference	73
5.21.1 Detailed Description	74
5.22 GeneratoryDanych.cpp File Reference	74
5.22.1 Detailed Description	75
5.22.2 Function Documentation	75
5.22.2.1 generujdane	75
5.22.2.2 generujdane	75
5.23 GeneratoryDanych.hh File Reference	75
5.23.1 Detailed Description	76
5.23.2 Function Documentation	76
5.23.2.1 generujdane	76
5.24 HaszTab.cpp File Reference	76
5.24.1 Detailed Description	77
5.25 HaszTab.hh File Reference	77
5.25.1 Detailed Description	77
5.26 Kolejka.cpp File Reference	78
5.26.1 Detailed Description	78
5.27 Kolejka.hh File Reference	78
5.27.1 Detailed Description	79

5.28	Lista.cpp File Reference	79
5.28.1	Detailed Description	80
5.29	Lista.hh File Reference	80
5.29.1	Detailed Description	81
5.30	main.cpp File Reference	81
5.30.1	Detailed Description	82
5.30.2	Function Documentation	82
5.30.2.1	main	82
5.31	Obserwator.cpp File Reference	82
5.31.1	Detailed Description	82
5.31.2	Function Documentation	82
5.31.2.1	odswiez	82
5.32	Obserwator.hh File Reference	82
5.32.1	Detailed Description	83
5.33	ObserwatorZapisujacy.cpp File Reference	83
5.33.1	Detailed Description	84
5.34	ObserwatorZapisujacy.hh File Reference	84
5.34.1	Detailed Description	85
5.35	Obserwowany.hh File Reference	85
5.35.1	Detailed Description	86
5.36	Stos.cpp File Reference	86
5.36.1	Detailed Description	87
5.37	Stos.hh File Reference	87
5.37.1	Detailed Description	88
5.38	Zasobnik.hh File Reference	88
5.38.1	Detailed Description	89
Index		91

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Algorytm< T >	7
Algorytm< int >	7
Algorytm1	8
Algorytm2	10
Algorytm4	13
Algorytm5	17
Algorytm6	20
Algorytm< string >	7
Algorytm3	11
Obserwator	45
ObserwatorZapisujacy	46
Obserwowany	47
Benchmark< T >	25
Zasobnik< T >	52
Zasobnik< int >	52
ArrayLista	22
DrzewoBinarne	27
DrzewoRB	31
Kolejka	40
Lista	42
Stos	50
Zasobnik< string >	52
HaszTab	35

Chapter 2

Class Index

2.1 Class List

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

Algorytm< T >	
Szablon klasy Algorytm	7
Algorytm1	
Klasa Algorytm1	8
Algorytm2	
Klasa Algorytm2	10
Algorytm3	
Klasa Algorytm3	11
Algorytm4	
Klasa Algorytm4	13
Algorytm5	
Klasa Algorytm5	17
Algorytm6	
Klasa Algorytm6	20
ArrayLista	
Klasa ArrayLista	22
Benchmark< T >	
Szablon klasy Benchmark	25
DrzewoBinarne	
Klasa DrzewoBinarne	27
DrzewoRB	
Klasa DrzewoRB	31
HaszTab	
Klasa HaszTab	35
Kolejka	
Klasa Kolejka	40
Lista	
Klasa Lista	42
Obserwator	
Klasa Obserwator	45
ObserwatorZapisujacy	
Klasa ObserwatorZapisujacy	46
Obserwowany	
Szablon klasy Obserwowany	47
Stos	
Klasa Stos	50
Zasobnik< T >	
Szablon klasy Zasobnik	52

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Algorytm1.cpp		
Metody klasy Algorytm1	55
Algorytm1.hh		
Definicja klasy Algorytm1	55
Algorytm2.cpp		
Metody klasy Algorytm2	56
Algorytm2.hh		
Definicja klasy Algorytm2	57
Algorytm3.cpp		
Metody klasy Algorytm3	58
Algorytm3.hh		
Definicja klasy Algorytm3	59
Algorytm4.cpp		
Metody klasy Algorytm4	60
Algorytm4.hh		
Definicja klasy Algorytm4	61
Algorytm5.cpp		
Metody klasy Algorytm5	62
Algorytm5.hh		
Definicja klasy Algorytm5	63
Algorytm6.cpp		
Metody klasy Algorytm6	64
Algorytm6.hh		
Definicja klasy Algorytm6	65
AlgorytmAbs.hh		
Definicja szablonu klasy abstrakcyjnej Algorytm	66
ArrayLista.cpp		
Metody klasy ArrayLista	67
ArrayLista.hh		
Definicja klasy ArrayLista	68
Benchmark.cpp		
Metody klasy Benchmark	69
Benchmark.hh		
Definicja szablonu klasy Benchmark	70
DrzewoBinarne.cpp		
Metody klasy DrzewoBinarne	71
DrzewoBinarne.hh		
Definicja klasy DrzewoBinarne	71

DrzewoRB.cpp	
Metody klasy DrzewoRB	72
DrzewoRB.hh	
Definicja klasy DrzewoRB	73
GeneratoryDanych.cpp	
Funkcje generacji danych	74
GeneratoryDanych.hh	
Szablon funkcji generacji danych	75
HaszTab.cpp	
Metody klasy HaszTab	76
HaszTab.hh	
Definicja klasy HaszTab	77
Kolejka.cpp	
Metody klasy Kolejka	78
Kolejka.hh	
Definicja klasy Kolejka	78
Lista.cpp	
Metody klasy Lista	79
Lista.hh	
Definicja klasy Lista	80
main.cpp	
Modul glowny	81
Obserwator.cpp	
Metody klasy Obserwator	82
Obserwator.hh	
Definicja klasy Obserwator	82
ObserwatorZapisujacy.cpp	
Metody klasy ObserwatorZapisujacy	83
ObserwatorZapisujacy.hh	
Definicja klasy ObserwatorZapisujacy	84
Obserwowany.hh	
Definicja szablonu klasy abstrakcyjnej Obserwowany	85
Stos.cpp	
Metody klasy Stos	86
Stos.hh	
Definicja klasy Stos	87
Zasobnik.hh	
Definicja szablonu klasy abstrakcyjnej Zasobnik	88

Chapter 4

Class Documentation

4.1 Algorytm< T > Class Template Reference

Szablon klasy [Algorytm](#).

```
#include <AlgorytmAbs.hh>
```

Public Member Functions

- virtual [~Algorytm](#) ()
Destruktor wirtualny algorytmu.
- virtual void [alokuj dane](#) ([Zasobnik](#)< T > *Tab, T *dane, int liczba_danych)=0
Metoda alokujaca na zasobniku dane.
- virtual void [wykonaj algorytm](#) ([Zasobnik](#)< T > *Tab, T *dane, int liczba_danych)=0
Metoda wykonujaca konkretny algorytm.

4.1.1 Detailed Description

```
template<typename T>class Algorytm< T >
```

Szablon klasy [Algorytm](#).

4.1.2 Constructor & Destructor Documentation

4.1.2.1 `template<typename T> virtual Algorytm< T >::~~Algorytm () [inline],[virtual]`

Destruktor wirtualny algorytmu.

4.1.3 Member Function Documentation

4.1.3.1 `template<typename T> virtual void Algorytm< T >::alokuj dane (Zasobnik< T > * Tab, T * dane, int liczba_danych) [pure virtual]`

Metoda alokujaca na zasobniku dane.

Parameters

<i>Tab</i>	- typu <code>Zasobnik<T>*</code> , implementacja zasobnika
<i>dane</i>	- typu <code>T*</code> , dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu <code>int</code> , liczba danych dla zasobnika

Implemented in [Algorytm1](#), [Algorytm2](#), [Algorytm3](#), [Algorytm4](#), [Algorytm5](#), and [Algorytm6](#).

4.1.3.2 `template<typename T> virtual void Algorytm< T >::wykonajalgorytm (Zasobnik< T > * Tab, T * dane, int liczba_danych) [pure virtual]`

Metoda wykonujaca konkretny algorytm.

Parameters

<i>Tab</i>	- typu <code>Zasobnik<T>*</code> , implementacja zasobnika
<i>dane</i>	- typu <code>T*</code> , dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu <code>int</code> , liczba danych dla zasobnika

Implemented in [Algorytm1](#), [Algorytm2](#), [Algorytm3](#), [Algorytm4](#), [Algorytm5](#), and [Algorytm6](#).

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

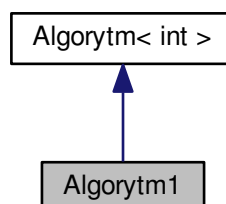
- [AlgorytmAbs.hh](#)

4.2 Algorytm1 Class Reference

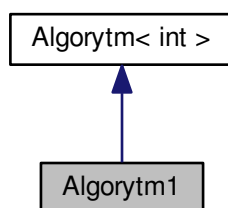
Klasa [Algorytm1](#).

```
#include <Algorytm1.hh>
```

Inheritance diagram for Algorytm1:



Collaboration diagram for Algorytm1:



Public Member Functions

- `~Algorytm1 ()`
- `void alokuj dane (Zasobnik< int > *, int *, int)`
Metoda alokująca na zasobniku dane.
- `void wykonaj algorytm (Zasobnik< int > *, int *, int)`
Metoda wykonująca konkretny algorytm.

4.2.1 Detailed Description

Klasa `Algorytm1`.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 `Algorytm1::~~Algorytm1 () [inline]`

4.2.3 Member Function Documentation

4.2.3.1 `void Algorytm1::alokuj dane (Zasobnik< int > * Tab, int * dane, int liczba_danych) [virtual]`

Metoda alokująca na zasobniku dane.

Parameters

<i>Tab</i>	- typu <code>Zasobnik<T>*</code> , implementacja zasobnika
<i>dane</i>	- typu <code>T*</code> , dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu <code>int</code> , liczba danych dla zasobnika

Implements `Algorytm< int >`.

4.2.3.2 `void Algorytm1::wykonaj algorytm (Zasobnik< int > * Tab, int * dane, int liczba_danych) [virtual]`

Metoda wykonująca konkretny algorytm.

Parameters

<i>Tab</i>	- typu <code>Zasobnik<T>*</code> , implementacja zasobnika
<i>dane</i>	- typu <code>T*</code> , dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu <code>int</code> , liczba danych dla zasobnika

Implements [Algorytm< int >](#).

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

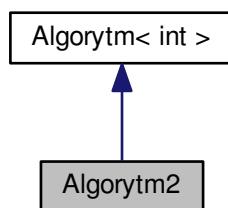
- [Algorytm1.hh](#)
- [Algorytm1.cpp](#)

4.3 Algorytm2 Class Reference

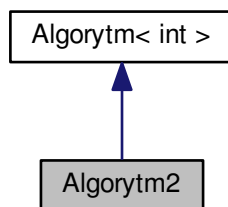
Klasa [Algorytm2](#).

```
#include <Algorytm2.hh>
```

Inheritance diagram for Algorytm2:



Collaboration diagram for Algorytm2:



Public Member Functions

- void [alokujdane](#) ([Zasobnik< int >*](#)Tab, int *dane, int liczba_danych)
Metoda alokująca na zasobniku dane.

- void [wykonajalgorytm](#) ([Zasobnik](#)< int > *Tab, int *dane, int liczba_danych)

Metoda wykonująca konkretny algorytm.

4.3.1 Detailed Description

Klasa [Algorytm2](#).

4.3.2 Member Function Documentation

4.3.2.1 void [Algorytm2::alokujdane](#) ([Zasobnik](#)< int > * Tab, int * dane, int liczba_danych) [virtual]

Metoda alokująca na zasobniku dane.

Parameters

<i>Tab</i>	- typu Zasobnik <T>*, implementacja zasobnika
<i>dane</i>	- typu T*, dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu int, liczba danych dla zasobnika

Implements [Algorytm](#)< int >.

4.3.2.2 void [Algorytm2::wykonajalgorytm](#) ([Zasobnik](#)< int > * Tab, int * dane, int liczba_danych) [virtual]

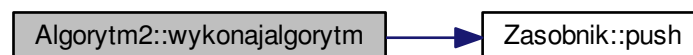
Metoda wykonująca konkretny algorytm.

Parameters

<i>Tab</i>	- typu Zasobnik <T>*, implementacja zasobnika
<i>dane</i>	- typu T*, dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu int, liczba danych dla zasobnika

Implements [Algorytm](#)< int >.

Here is the call graph for this function:



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

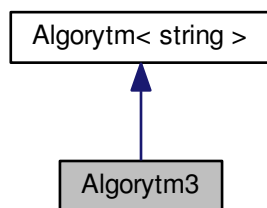
- [Algorytm2.hh](#)
- [Algorytm2.cpp](#)

4.4 Algorytm3 Class Reference

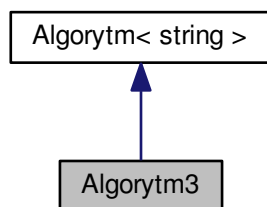
Klasa [Algorytm3](#).

```
#include <Algorytm3.hh>
```

Inheritance diagram for Algorytm3:



Collaboration diagram for Algorytm3:



Public Member Functions

- void [alokujdane](#) ([Zasobnik](#)< string > *Tab, string *dane, int liczba_danych)
Metoda alokujaca na zasobniku dane.
- void [wykonajalgorytm](#) ([Zasobnik](#)< string > *Tab, string *dane, int liczba_danych)
Metoda wykonujaca konkretny algorytm.

4.4.1 Detailed Description

Klasa [Algorytm3](#).

4.4.2 Member Function Documentation

4.4.2.1 void `Algorytm3::alokujdane (Zasobnik< string > * Tab, string * dane, int liczba_danych)` `[virtual]`

Metoda alokujaca na zasobniku dane.

Parameters

<i>Tab</i>	- typu <code>Zasobnik<T>*</code> , implementacja zasobnika
<i>dane</i>	- typu <code>T*</code> , dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu <code>int</code> , liczba danych dla zasobnika

Implements [Algorytm< string >](#).

Here is the call graph for this function:



4.4.2.2 `void Algorytm3::wykonajalgorytm (Zasobnik< string > * Tab, string * dane, int liczba_danych) [virtual]`

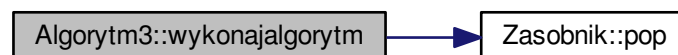
Metoda wykonująca konkretny algorytm.

Parameters

<i>Tab</i>	- typu <code>Zasobnik<T>*</code> , implementacja zasobnika
<i>dane</i>	- typu <code>T*</code> , dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu <code>int</code> , liczba danych dla zasobnika

Implements [Algorytm< string >](#).

Here is the call graph for this function:



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

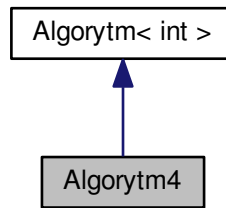
- [Algorytm3.hh](#)
- [Algorytm3.cpp](#)

4.5 Algorytm4 Class Reference

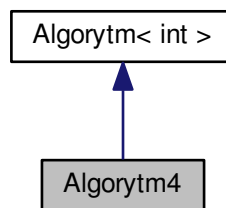
Klasa [Algorytm4](#).

```
#include <Algorytm4.hh>
```

Inheritance diagram for Algorytm4:



Collaboration diagram for Algorytm4:



Public Member Functions

- [~Algorytm4](#) ()
- void [alokujdane](#) ([Zasobnik](#)< int > *, int *, int)
- void [wykonajalgorytm](#) ([Zasobnik](#)< int > *, int *, int)
- int * [mergesort](#) (int *, int)
- int * [scal](#) (int *, int *, int, int *, int)

4.5.1 Detailed Description

Klasa [Algorytm4](#).

4.5.2 Constructor & Destructor Documentation

4.5.2.1 [Algorytm4::~Algorytm4](#) () [inline]

4.5.3 Member Function Documentation

4.5.3.1 void Algorytm4::alokuj dane (Zasobnik< int > * Tab, int * dane, int liczba_danych) [virtual]

Metoda alokująca na zasobniku dane.

Parameters

<i>Tab</i>	- typu <code>Zasobnik<T>*</code> , implementacja zasobnika
<i>dane</i>	- typu <code>T*</code> , dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu <code>int</code> , liczba danych dla zasobnika

Implements [Algorytm< int >](#).

Here is the call graph for this function:



4.5.3.2 `int * Algorytm4::mergesort (int * Tab, int wielkosc)`

Here is the call graph for this function:



4.5.3.3 `int * Algorytm4::scal (int * Tab, int * tab_l, int size_l, int * tab_p, int size_p)`

4.5.3.4 `void Algorytm4::wykonajalgorytm (Zasobnik< int > * Tab, int * dane, int liczba_danych)` [virtual]

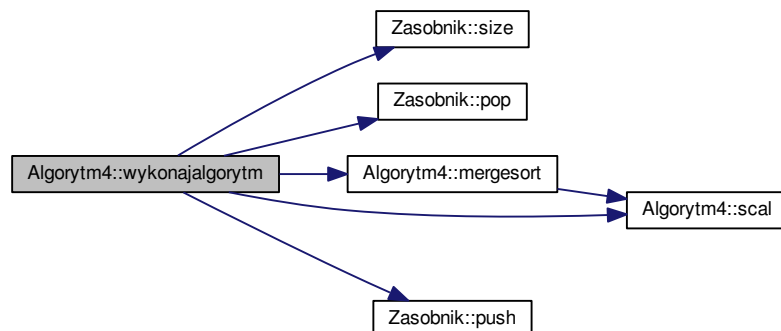
Metoda wykonująca konkretny algorytm.

Parameters

<i>Tab</i>	- typu <code>Zasobnik<T>*</code> , implementacja zasobnika
<i>dane</i>	- typu <code>T*</code> , dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu <code>int</code> , liczba danych dla zasobnika

Implements [Algorytm< int >](#).

Here is the call graph for this function:



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

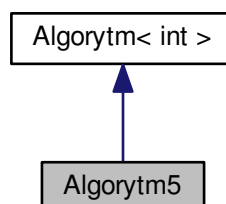
- [Algorytm4.hh](#)
- [Algorytm4.cpp](#)

4.6 Algorytm5 Class Reference

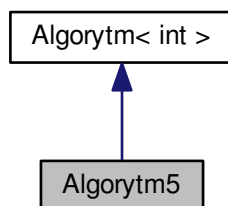
Klasa [Algorytm5](#).

```
#include <Algorytm5.hh>
```

Inheritance diagram for Algorytm5:



Collaboration diagram for Algorytm5:



Public Member Functions

- void [alokujdane](#) ([Zasobnik<int> *](#), int *, int)
Metoda alokujaca na zasobniku dane.
- void [wykonajalgorytm](#) ([Zasobnik<int> *](#), int *, int)
Metoda wykonujaca konkretny algorytm.

4.6.1 Detailed Description

Klasa [Algorytm5](#).

4.6.2 Member Function Documentation

4.6.2.1 void `Algorytm5::alokujdane (Zasobnik<int> * Tab, int * dane, int liczba_danych)` [virtual]

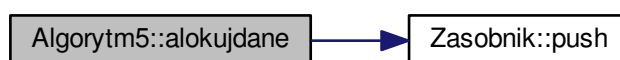
Metoda alokujaca na zasobniku dane.

Parameters

<i>Tab</i>	- typu <code>Zasobnik<T>*</code> , implementacja zasobnika
<i>dane</i>	- typu <code>T*</code> , dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu <code>int</code> , liczba danych dla zasobnika

Implements [Algorytm<int>](#).

Here is the call graph for this function:



4.6.2.2 void Algorytm5::wykonajalgorytm (Zasobnik< int > * *Tab*, int * *dane*, int *liczba_danych*) [virtual]

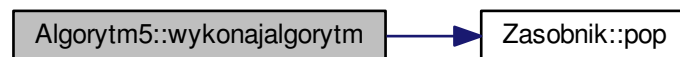
Metoda wykonujaca konkretny algorytm.

Parameters

<i>Tab</i>	- typu <code>Zasobnik<T>*</code> , implementacja zasobnika
<i>dane</i>	- typu <code>T*</code> , dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu <code>int</code> , liczba danych dla zasobnika

Implements [Algorytm< int >](#).

Here is the call graph for this function:



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

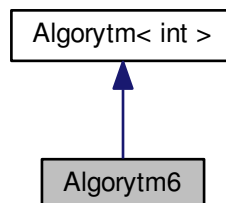
- [Algorytm5.hh](#)
- [Algorytm5.cpp](#)

4.7 Algorytm6 Class Reference

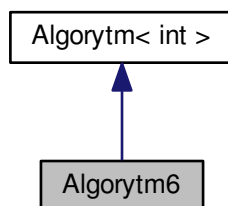
Klasa [Algorytm6](#).

```
#include <Algorytm6.hh>
```

Inheritance diagram for Algorytm6:



Collaboration diagram for Algorytm6:



Public Member Functions

- `~Algorytm6()`
- `void alokuj dane (Zasobnik<int> *, int *, int)`
Metoda alokujaca na zasobniku dane.
- `void wykonajalgorytm (Zasobnik<int> *, int *, int)`
Metoda wykonujaca konkretny algorytm.

4.7.1 Detailed Description

Klasa `Algorytm6`.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 `Algorytm6::~~Algorytm6 ()` `[inline]`

4.7.3 Member Function Documentation

4.7.3.1 `void Algorytm6::alokuj dane (Zasobnik<int> * Tab, int * dane, int liczba_danych)` `[virtual]`

Metoda alokujaca na zasobniku dane.

Parameters

<i>Tab</i>	- typu <code>Zasobnik<T>*</code> , implementacja zasobnika
<i>dane</i>	- typu <code>T*</code> , dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu <code>int</code> , liczba danych dla zasobnika

Implements `Algorytm<int>`.

Here is the call graph for this function:



4.7.3.2 void Algoritm6::wykonajalgorytm (Zasobnik< int > * Tab, int * dane, int liczba_danych) [virtual]

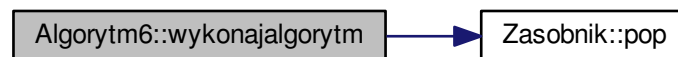
Metoda wykonująca konkretny algorytm.

Parameters

<i>Tab</i>	- typu Zasobnik<T>*, implementacja zasobnika
<i>dane</i>	- typu T*, dane wygenerowane dla implementacji
<i>liczba_danych</i>	- typu int, liczba danych dla zasobnika

Implements [Algoritm< int >](#).

Here is the call graph for this function:



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

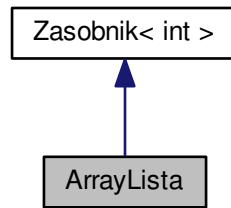
- [Algoritm6.hh](#)
- [Algoritm6.cpp](#)

4.8 ArrayLista Class Reference

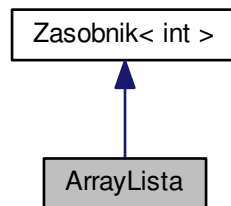
Klasa [ArrayLista](#).

```
#include <ArrayLista.hh>
```


Inheritance diagram for ArrayLista:



Collaboration diagram for ArrayLista:



Public Member Functions

- [ArrayLista](#) ()

Konstruktor bezparametryczny. Konstruktor inicjalizujący tablice listy. rozmiar - rozmiar tablicy dla listy, inicjalizowane wartoscia 1. ilosc_elementow - ilosc elementow listy, inicjalizowane wartoscia 0. Tworzona jest tablica dynamiczna o jednym elemencie.

- [ArrayLista](#) (int)

- [~ArrayLista](#) ()

Destruktor bezparametryczny Listy.

- void [push](#) (int, int)

Metoda umieszczająca element na liście. Metoda inkrementuje rozmiar podczas umieszczania elementu na liście.

- void [push](#) (int wartosc)

Przeciążenie operacji push. Powiększa listę podczas dodawania elementu do 200%. Następuje inkrementacja rozmiaru listy.

- int [pop](#) ()

Metoda zdejmująca element z listy. Metoda dekrementuje ilosc_elementow przy zdejmowaniu z listy. Tablica listy jest zmniejszana podczas zdejmowania elementu o połowę gdy ilosc elementow znajdujących się na niej jest równa połowie jej rozmiaru.

- int [size](#) ()

Metoda zwracająca rozmiar tablicy na której oparta jest lista.

4.8.1 Detailed Description

Klasa [ArrayLista](#).

4.8.2 Constructor & Destructor Documentation

4.8.2.1 ArrayLista::ArrayLista ()

Konstruktor bezparametryczny. Konstruktor inicjalizujący tablice listy. *rozmiar* - rozmiar tablicy dla listy, inicjalizowane wartością 1. *ilosc_elementow* - ilość elementów listy, inicjalizowane wartością 0. Tworzona jest tablica dynamiczna o jednym elemencie.

4.8.2.2 ArrayLista::ArrayLista (int *wielkosc*)

4.8.2.3 ArrayLista::~~ArrayLista ()

Destruktor bezparametryczny Listy.

4.8.3 Member Function Documentation

4.8.3.1 int ArrayLista::pop () [virtual]

Metoda zdejmująca element z listy. Metoda dekrementuje *ilosc_elementow* przy zdejmowaniu z listy. Tablica listy jest zmniejszana podczas zdejmowania elementu o połowę gdy ilość elementów znajdujących się na niej jest równa połowie jej rozmiaru.

Returns

wartosc - typu int, wartosc zdejmowana ze stosu.

Implements [Zasobnik< int >](#).

4.8.3.2 void ArrayLista::push (int *wartosc*, int *zwiększanie*)

Metoda umieszczająca element na liście Metoda inkrementuje rozmiar podczas umieszczania elementu na liście.

Parameters

<i>zwiększanie</i>	- typu int, mnożnik rozszerzania tablicy podczas dodawania elementów listy .
<i>wartosc</i>	- typu int, wartosc umieszczana na stosie.

4.8.3.3 void ArrayLista::push (int *wartosc*) [inline],[virtual]

Przeciążenie operacji push. Powiększa listę podczas dodawania elementu do 200%. Następuje inkrementacja rozmiaru listy.

Parameters

<i>wartosc</i>	- typu int, wartosc umieszczana na liście.
----------------	--

Implements [Zasobnik< int >](#).

4.8.3.4 int ArrayLista::size () [virtual]

Metoda zwracająca rozmiar tablicy na której oparta jest lista.

Returns

rozmiar - typu int, rozmiar tablicy listy.

Implements [Zasobnik< int >](#).

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

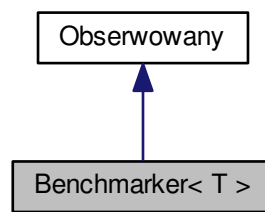
- [ArrayLista.hh](#)
- [ArrayLista.cpp](#)

4.9 Benchmark< T > Class Template Reference

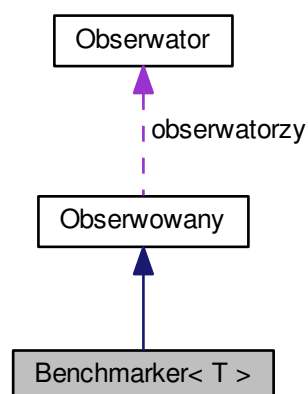
Szablon klasy [Benchmark](#).

```
#include <Benchmark.hh>
```

Inheritance diagram for Benchmark< T >:



Collaboration diagram for Benchmark< T >:



Public Member Functions

- void [testuj](#) ([Zasobnik](#)< T > *, [Algorytm](#)< T > *, T *, int, int)
Szablon metody przeprowadzająca sprawdzenie czasu działania funkcji. Typy: [Lista](#) , [Stos](#) , [Kolejka](#) , [HaszTab](#).
- void [powiadom](#) (int, long int)
Metoda powiadamiająca obserwatora o czasie wykonania.

Additional Inherited Members

4.9.1 Detailed Description

template<typename T>class [Benchmark](#)< T >

Szablon klasy [Benchmark](#).

4.9.2 Member Function Documentation

4.9.2.1 template<typename T > void [Benchmark](#)< T >::powiadom (int *iteracja*, long int *czas_sredni*)

Metoda powiadamiająca obserwatora o czasie wykonania.

Parameters

<i>iteracja</i>	- typu int, liczba danych - identyfikator iteracji
<i>czas_sredni</i>	- typu long int, czas wykonania operacji

4.9.2.2 template<typename T > template void [Benchmark](#)< T >::testuj ([Zasobnik](#)< T > *, [Algorytm](#)< T > *, T *, int, int)

Szablon metody przeprowadzająca sprawdzenie czasu działania funkcji. Typy: [Lista](#) , [Stos](#) , [Kolejka](#) , [HaszTab](#).

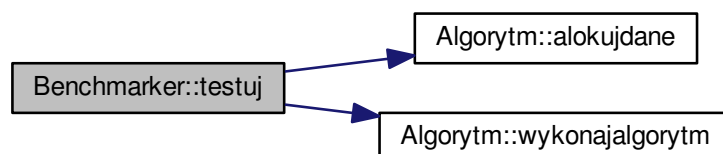
Template Parameters

<i>Tab</i>	- typu T*, wskaznik na zaimplementowany stos/liste/kolejke/tablice haszującą.
<i>dane</i>	- typu int*, wskaznik na tablice z danymi generowanymi.
<i>liczba_przejsc</i>	- typu int, liczba przejsc przez dane.
<i>liczba_danych</i>	- typu int, liczba danych w tablicy.

Returns

czas_calkowity_usredniony - typu long int, czas sredni działania funkcji.

Here is the call graph for this function:



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

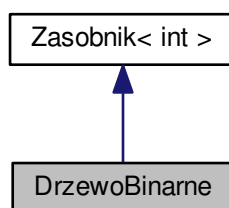
- [Benchmark.hh](#)
- [Benchmark.cpp](#)

4.10 DrzewoBinarne Class Reference

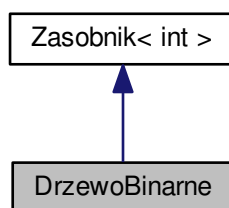
Klasa [DrzewoBinarne](#).

```
#include <DrzewoBinarne.hh>
```

Inheritance diagram for DrzewoBinarne:



Collaboration diagram for DrzewoBinarne:



Public Member Functions

- [DrzewoBinarne](#) ()
*Konstruktor bezparametryczny. Konstruktor inicjalizujący drzewo binarne. korzen - inicjalizowany wartoscia NULL
rozmiar - inicjalizowany wartoscia 0.*
- [~DrzewoBinarne](#) ()
Destruktor bezparametryczny drzewa binarnego.
- void [push](#) (int wartosc)
Metoda umieszczająca element na drzewie binarnym.
- int [pop](#) ()

- Metoda zdejmujaca element z drzewa binarnego.*
- int [pop](#) (int szukana)
- Metoda zdejmujaca dany element z drzewa binarnego.*
- int [size](#) ()
- Metoda zwracajaca ilosc wezlow drzewa binarnego.*
- void [wypisz_pelne](#) ()
- Metoda wypisujaca cale drzewo. Wywoluje kolejne wypisz dla lewego oraz prawego subdrzewa korzenia.*
- void [wypisz](#) (element *iter, int a)
- Metoda wypisujaca subdrzewo.*
- element * [obroc_l](#) (element *pivot)
- Metoda obracajaca w lewo dla elementu.*
- element * [obroc_p](#) (element *pivot)
- Metoda obracajaca w prawo dla elementu.*
- element * [rownowaz](#) (element *tmp)
- Metoda rownowazaca drzewo binarne.*

4.10.1 Detailed Description

Klasa [DrzewoBinarne](#).

4.10.2 Constructor & Destructor Documentation

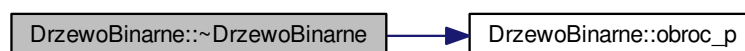
4.10.2.1 DrzewoBinarne::DrzewoBinarne ()

Konstruktor bezparametryczny. Konstruktor inicjalizujący drzewo binarne. korzen - inicjalizowany wartoscia NULL
rozmiar - inicjalizowany wartoscia 0.

4.10.2.2 DrzewoBinarne::~~DrzewoBinarne ()

Destruktor bezparametryczny drzewa binarnego.

Here is the call graph for this function:



4.10.3 Member Function Documentation

4.10.3.1 DrzewoBinarne::element * DrzewoBinarne::obroc_l (element * pivot)

Metoda obracajaca w lewo dla elementu.

Parameters

<i>pivot</i>	- typu element* , zadany wskaznik na element , dla ktorego wystapi obrocenie.
--------------	---

Returns

pivot - typu element* , wskaznik na element , ktory wskoczył za dany.

4.10.3.2 DrzewoBinarne::element * DrzewoBinarne::obroc_p (element * pivot)

Metoda obracająca w prawo dla elementu.

Parameters

<i>pivot</i>	- typu element* , zadany wskaznik na element , dla ktorego wystapi obrocenie.
--------------	---

Returns

pivot - typu element* , wskaznik na element , ktory wskoczył za dany.

4.10.3.3 int DrzewoBinarne::pop () [virtual]

Metoda zdejmująca element z drzewa binarnego.

Returns

wartosc - typu int, wartosc zdejmowana z drzewa binarnego.

Implements [Zasobnik< int >](#).

4.10.3.4 int DrzewoBinarne::pop (int szukana) [virtual]

Metoda zdejmująca dany element z drzewa binarnego.

Parameters

<i>szukana</i>	- typu int, wartosc szukana na drzewie binarnym.
----------------	--

Returns

wartosc - typu int, wartosc zdejmowana z drzewa binarnego.

Implements [Zasobnik< int >](#).

4.10.3.5 void DrzewoBinarne::push (int wartosc) [virtual]

Metoda umieszczająca element na drzewie binarnym.

Parameters

<i>wartosc</i>	- typu int, wartosc umieszczana na drzewie binarnym.
----------------	--

Implements [Zasobnik< int >](#).

Here is the call graph for this function:



4.10.3.6 `DrzewoBinarne::element * DrzewoBinarne::rownowaz (element * tmp)`

Metoda rownowazaca drzewo binarne.

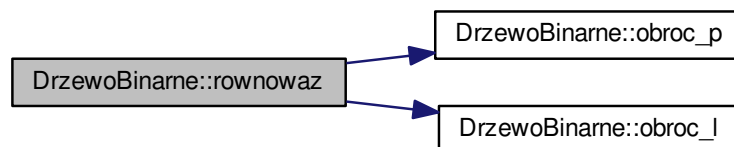
Parameters

<i>tmp</i>	- typu <code>element*</code> , zadany wskaznik , najczesciej korzen.
------------	--

Returns

tmp - typu `element*`, wskaznik na korzen.

Here is the call graph for this function:



4.10.3.7 `int DrzewoBinarne::size () [virtual]`

Metoda zwracajaca ilosc wezlow drzewa binarnego.

Returns

rozmiar - typu `int`, ilosc wezlow drzewa binarnego.

Implements [Zasobnik< int >](#).

4.10.3.8 `void DrzewoBinarne::wypisz (element * iter, int a)`

Metoda wypisujaca subdrzewo.

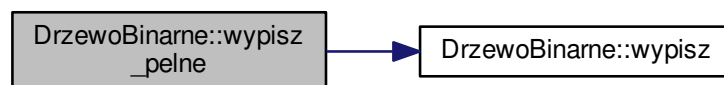
Parameters

<i>iter</i>	- typu element* , zadany wskaznik na subdrzewo.
<i>a</i>	- typu int, poziom wysisywanego subdrzewa.

4.10.3.9 void DrzewoBinarne::wypisz_pelne ()

Metoda wypisujaca cale drzewo. Wywoluje kolejne wypisz dla lewego oraz prawego subdrzewa korzenia.

Here is the call graph for this function:



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

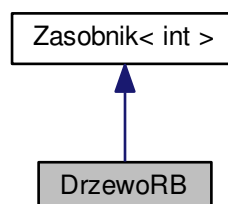
- [DrzewoBinarne.hh](#)
- [DrzewoBinarne.cpp](#)

4.11 DrzewoRB Class Reference

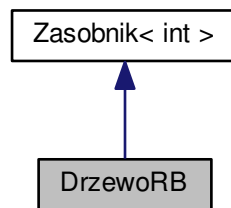
Klasa [DrzewoRB](#).

```
#include <DrzewoRB.hh>
```

Inheritance diagram for DrzewoRB:



Collaboration diagram for DrzewoRB:



Public Member Functions

- [DrzewoRB](#) ()
*Konstruktor bezparametryczny. Konstruktor inicjalizujący drzewo binarne. korzen - inicjalizowany wartoscia NULL
rozmiar - inicjalizowany wartoscia 0.*
- [~DrzewoRB](#) ()
Destruktor bezparametryczny drzewa RB.
- void [push](#) (int wartosc)
Metoda umieszczająca element na drzewie RB.
- int [pop](#) ()
Metoda zdejmująca element z drzewa RB.
- int [pop](#) (int szukana)
Metoda zdejmująca dany element z drzewa RB.
- int [size](#) ()
Metoda zwracająca ilość węzłów drzewa RB.
- void [wypisz_pelne](#) ()
Metoda wypisująca całe drzewo. Wywołuje kolejne wypisz dla lewego oraz prawego subdrzewa korzenia.
- void [wypisz](#) (element *iter, int a)
Metoda wypisująca subdrzewo.
- element * [obroc_l](#) (element *pivot)
Metoda obracająca w lewo dla elementu.
- element * [obroc_p](#) (element *pivot)
Metoda obracająca w prawo dla elementu.
- element * [ukladaj](#) (element *pivot)
Metoda układająca element na drzewie RB.

4.11.1 Detailed Description

Klasa [DrzewoRB](#).

4.11.2 Constructor & Destructor Documentation

4.11.2.1 DrzewoRB::DrzewoRB ()

Konstruktor bezparametryczny. Konstruktor inicjalizujący drzewo binarne. korzen - inicjalizowany wartoscia NULL
rozmiar - inicjalizowany wartoscia 0.

4.11.2.2 DrzewoRB::~~DrzewoRB ()

Destruktor bezparametryczny drzewa RB.

Here is the call graph for this function:



4.11.3 Member Function Documentation

4.11.3.1 DrzewoRB::element * DrzewoRB::obroc_l (element * *pivot*)

Metoda obracająca w lewo dla elementu.

Parameters

<i>pivot</i>	- typu element*, zadany wskaźnik na element , dla ktorego wystąpi obrocenie.
--------------	--

Returns

pivot - typu element*, wskaźnik na element , który wskoczył za dany.

4.11.3.2 DrzewoRB::element * DrzewoRB::obroc_p (element * *pivot*)

Metoda obracająca w prawo dla elementu.

Parameters

<i>pivot</i>	- typu element*, zadany wskaźnik na element , dla ktorego wystąpi obrocenie.
--------------	--

Returns

pivot - typu element*, wskaźnik na element , który wskoczył za dany.

4.11.3.3 int DrzewoRB::pop () [virtual]

Metoda zdejmująca element z drzewa RB.

Returns

wartosc - typu int, wartosc zdejmowana z drzewa RB.

Implements [Zasobnik< int >](#).

4.11.3.4 int DrzewoRB::pop (int *szukana*) [virtual]

Metoda zdejmująca dany element z drzewa RB.

Parameters

<i>szukana</i>	- typu int, wartosc szukana na drzewie RB.
----------------	--

Returns

wartosc - typu int, wartosc zdejmowana z drzewa RB.

Implements [Zasobnik< int >](#).

4.11.3.5 void DrzewoRB::push (int *wartosc*) [virtual]

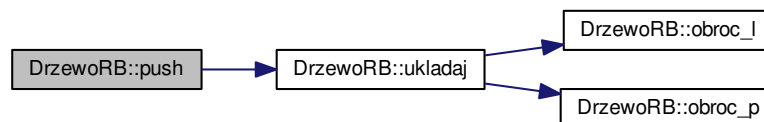
Metoda umieszczajaca element na drzewie RB.

Parameters

<i>wartosc</i>	- typu int, wartosc umieszczana na drzewie RB.
----------------	--

Implements [Zasobnik< int >](#).

Here is the call graph for this function:



4.11.3.6 int DrzewoRB::size () [virtual]

Metoda zwracajaca ilosc wezlow drzewa RB.

Returns

rozmiar - typu int, ilosc wezlow drzewa RB.

Implements [Zasobnik< int >](#).

4.11.3.7 DrzewoRB::element * DrzewoRB::ukladaj (element * *pivot*)

Metoda ukkladajaca element na drzewie RB.

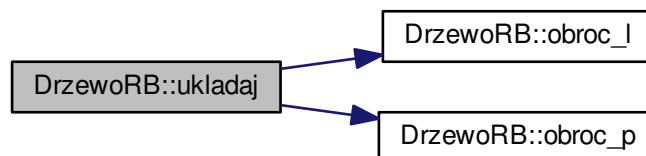
Parameters

<i>pivot</i>	- typu element*, zadany wskaznik , najczesciej korzen.
--------------	--

Returns

pivot - typu element*, wskaznik na element.

Here is the call graph for this function:



4.11.3.8 void DrzewoRB::wypisz (element * iter, int a)

Metoda wypisujaca subdrzewo.

Parameters

<i>iter</i>	- typu element* , zadany wskaznik na subdrzewo.
<i>a</i>	- typu int, poziom wysisywanego subdrzewa.

4.11.3.9 void DrzewoRB::wypisz_pelne ()

Metoda wypisujaca cale drzewo. Wywoluje kolejne wypisz dla lewego oraz prawego subdrzewa korzenia.

Here is the call graph for this function:



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

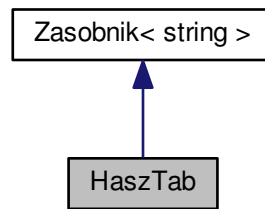
- [DrzewoRB.hh](#)
- [DrzewoRB.cpp](#)

4.12 HaszTab Class Reference

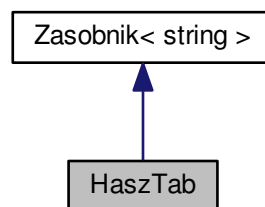
Klasa [HaszTab](#).

```
#include <HaszTab.hh>
```

Inheritance diagram for HaszTab:



Collaboration diagram for HaszTab:



Public Member Functions

- [HaszTab](#) ()
Konstruktor bezparametryczny. Konstruktor inicjalizujący tablice haszującą. rozmiar_k1 - wielkosc tablicy pierwszej. rozmiar_k2 - wielkosc tablicy drugiej. Tworzona jest macierz elementow (string wartosc,string klucz) o zadanej wielkości.
- [~HaszTab](#) ()
Destruktor bezparametryczny tablicy haszującej.
- void [push](#) (string wartosc, string klucz)
Przeciążenie operacji push. Metoda umieszczająca element o zadanej wartości i kluczu na tablicy haszującej.
- string [pop](#) (string klucz_szukany)
Metoda zdejmująca element z tablicy haszującej.
- void [push](#) (string wartosc)
Metoda umieszczająca element na tablicy haszującej.
- string [pop](#) ()
Metoda zdejmująca element z tablicy haszującej.
- int [size](#) ()
Metoda zwracająca rozmiar Tablicy Haszującej.
- int [mieszaj](#) (string klucz_umieszczany, int modulacja)
Metoda mieszająca klucz tablicy haszującej. Metoda zmienia zadany klucz na indeksy tablic.
- string [odczytaj](#) (string klucz_szukany)

Metoda czytująca element z tablicy haszującej. Metoda czytuje element o określonym kluczu. Wartości oraz klucze nie są usuwane podczas czytania.

- int [size_k1](#) ()

Metoda zwracająca rozmiar tablicy pierwszej na której oparta jest tablica haszująca.

- int [size_k2](#) ()

Metoda zwracająca rozmiar tablicy drugiej na której oparta jest tablica haszująca.

4.12.1 Detailed Description

Klasa [HaszTab](#).

4.12.2 Constructor & Destructor Documentation

4.12.2.1 HaszTab::HaszTab ()

Konstruktor bezparametryczny. Konstruktor inicjalizujący tablice haszującą. rozmiar_k1 - wielkość tablicy pierwszej. rozmiar_k2 - wielkość tablicy drugiej. Tworzona jest macierz elementów (string wartość,string klucz) o zadanej wielkości.

4.12.2.2 HaszTab::~~HaszTab ()

Destruktor bezparametryczny tablicy haszującej.

4.12.3 Member Function Documentation

4.12.3.1 int HaszTab::mieszaj (string klucz_umieszczany, int modulacja)

Metoda mieszająca klucz tablicy haszującej. Metoda zmienia zadany klucz na indeksy tablic.

Parameters

<i>klucz_↔ umieszczany</i>	- typu string, zadany klucz.
<i>modulacja</i>	- typu int, wielkość tablicy która moduluje klucz.

Returns

indeks - typu int, zmodulowany indeks elementu.

4.12.3.2 string HaszTab::odczytaj (string klucz_szukany)

Metoda czytująca element z tablicy haszującej. Metoda czytuje element o określonym kluczu. Wartości oraz klucze nie są usuwane podczas czytania.

Parameters

<i>klucz_szukany</i>	- typu string, szukany klucz.
----------------------	-------------------------------

Returns

wartosc - typu int, wartosc zdejmowana ze stosu.

Here is the call graph for this function:

**4.12.3.3** `string HaszTab::pop (string klucz_szukany) [virtual]`

Metoda zdejmująca element z tablicy haszującej.

Returns

klucz_szukany - typu string, klucz wartosci zdejmowanej z tablicy haszującej.

Implements [Zasobnik< string >](#).

4.12.3.4 `string HaszTab::pop () [virtual]`

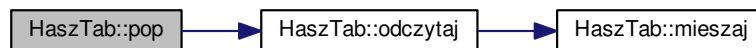
Metoda zdejmująca element z tablicy haszującej.

Returns

wartosc - typu string, wartosc zdejmowana z tablicy haszującej.

Implements [Zasobnik< string >](#).

Here is the call graph for this function:

**4.12.3.5** `void HaszTab::push (string wartosc, string klucz)`

Przeciążenie operacji push. Metoda umieszczająca element o zadanej wartości i kluczu na tablicy haszującej.

Parameters

<i>wartosc</i>	- typu string, zadana wartość.
<i>klucz</i>	- typu string, zadany klucz.

Here is the call graph for this function:

4.12.3.6 void HaszTab::push (string *wartosc*) [virtual]

Metoda umieszczająca element na tablicy haszującej.

Parameters

<i>wartosc</i>	- typu string, wartosc umieszczana na tablicy haszującej.
----------------	---

Implements [Zasobnik< string >](#).

4.12.3.7 int HaszTab::size () [virtual]

Metoda zwracająca rozmiar Tablicy Haszującej.

Returns

rozmiar - typu int, rozmiar Tablicy Haszującej.

Implements [Zasobnik< string >](#).

4.12.3.8 int HaszTab::size_k1 ()

Metoda zwracająca rozmiar tablicy pierwszej na której oparta jest tablica haszująca.

Returns

rozmiar - typu int, rozmiar tablicy pierwszej.

4.12.3.9 int HaszTab::size_k2 ()

Metoda zwracająca rozmiar tablicy drugiej na której oparta jest tablica haszująca.

Returns

rozmiar - typu int, rozmiar tablicy drugiej.

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

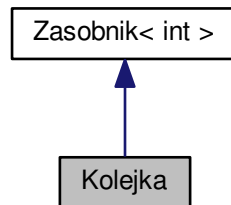
- [HaszTab.hh](#)
- [HaszTab.cpp](#)

4.13 Kolejka Class Reference

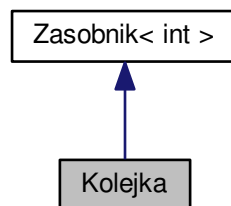
Klasa [Kolejka](#).

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

Inheritance diagram for Kolejka:



Collaboration diagram for Kolejka:



Public Member Functions

- [Kolejka](#) ()
Konstruktor bezparametryczny. Konstruktor inicjalizujący strażnika_początek i strażnika_koniec kolejki wartościami NULL, oraz rozmiar kolejki wartością 0.
- [~Kolejka](#) ()
Destruktor bezparametryczny kolejki.
- void [push](#) (int wartosc)
Metoda umieszczająca element na końcu kolejki. Metoda inkrementuje rozmiar podczas umieszczania elementu w kolejce.
- int [pop](#) ()
Metoda zdejmująca element z początku kolejki. Metoda dekrementuje rozmiar przy zdejmowaniu elementu.
- int [pop](#) (int a)
Metoda zdejmująca element z początku kolejki. Metoda dekrementuje rozmiar przy zdejmowaniu elementu.
- int [size](#) ()
Metoda zwracająca wielkość kolejki.

4.13.1 Detailed Description

Klasa [Kolejka](#).

4.13.2 Constructor & Destructor Documentation

4.13.2.1 Kolejka::Kolejka ()

Konstruktor bezparametryczny. Konstruktor inicjalizujący strażnika_początek i strażnika_koniec kolejki wartościami NULL , oraz rozmiar kolejki wartością 0.

4.13.2.2 Kolejka::~~Kolejka ()

Destruktor bezparametryczny kolejki.

4.13.3 Member Function Documentation

4.13.3.1 int Kolejka::pop () [virtual]

Metoda zdejmująca element z początku kolejki. Metoda dekrementuje rozmiar przy zdejmowaniu elementu.

Returns

wartosc - typu int, wartosc zdejmowana z kolejki.

Implements [Zasobnik< int >](#).

4.13.3.2 int Kolejka::pop (int a) [inline],[virtual]

Metoda zdejmująca element z początku kolejki. Metoda dekrementuje rozmiar przy zdejmowaniu elementu.

Returns

wartosc - typu int, wartosc zdejmowana z kolejki.

Implements [Zasobnik< int >](#).

4.13.3.3 void Kolejka::push (int wartosc) [virtual]

Metoda umieszczająca element na końcu kolejki. Metoda inkrementuje rozmiar podczas umieszczania elementu w kolejce.

Parameters

<i>wartosc</i>	- typu int, wartosc umieszczana na koncu kolejki.
----------------	---

Implements [Zasobnik< int >](#).

4.13.3.4 int Kolejka::size () [virtual]

Metoda zwracająca wielkość kolejki.

Returns

rozmiar - typu int, rozmiar kolejki.

Implements [Zasobnik< int >](#).

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

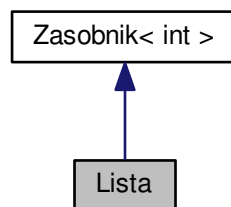
- [Kolejka.hh](#)
- [Kolejka.cpp](#)

4.14 Lista Class Reference

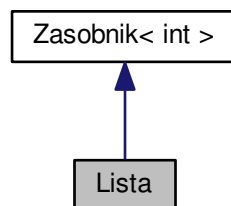
Klasa [Lista](#).

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

Inheritance diagram for Lista:



Collaboration diagram for Lista:



Public Member Functions

- [Lista](#) ()
Konstruktor bezparametryczny. Konstruktor inicjalizujący strażnika listy wartością NULL oraz rozmiar wartością 0.
- [~Lista](#) ()
Destruktor bezparametryczny listy.

- void [push](#) (int, int)
Metoda umieszczająca element określonej pozycji na liście <0,rozmiar>. Metoda inkrementuje rozmiar podczas umieszczania elementu na liście.
- int [pop](#) (int)
Metoda zdejmująca element z określonej pozycji listy <0,rozmiar>. Metoda dekrementuje rozmiar przy zdejmowaniu elementu.
- void [push](#) (int wartosc)
Przeciążenie operacji push. Umieszcza element domyślnie na pozycji 1. Następuje inkrementacja rozmiaru listy.
- int [pop](#) ()
Przeciążenie operacji pop dla listy. Pobiera domyślnie element listy z pozycji 1. Następuje dekrementacja rozmiaru listy.
- int [size](#) ()
Metoda zwracająca wielkość listy.

4.14.1 Detailed Description

Klasa [Lista](#).

4.14.2 Constructor & Destructor Documentation

4.14.2.1 `Lista::Lista ()`

Konstruktor bezparametryczny. Konstruktor inicjalizujący strażnika listy wartością NULL oraz rozmiar wartością 0.

4.14.2.2 `Lista::~~Lista ()`

Destruktor bezparametryczny listy.

4.14.3 Member Function Documentation

4.14.3.1 `int Lista::pop (int pozycja) [virtual]`

Metoda zdejmująca element z określonej pozycji listy <0,rozmiar>. Metoda dekrementuje rozmiar przy zdejmowaniu elementu.

Parameters

<i>pozycja</i>	- typu int, numer elementu który ma być zdjęty z listy.
----------------	---

Returns

wartosc - typu int, wartosc zdejmowana z listy.

Implements [Zasobnik< int >](#).

4.14.3.2 `int Lista::pop () [inline],[virtual]`

Przeciążenie operacji pop dla listy. Pobiera domyślnie element listy z pozycji 1. Następuje dekrementacja rozmiaru listy.

Returns

wartosc - typu int, wartosc zdejmowana z listy.

Implements [Zasobnik< int >](#).

4.14.3.3 void Lista::push (int *wartosc*, int *pozycja*)

Metoda umieszczająca element określonej pozycji na liście <0,rozmiar>. Metoda inkrementuje rozmiar podczas umieszczania elementu na liście.

Parameters

<i>wartosc</i>	- typu int, wartosc umieszczana na liscie.
<i>pozycja</i>	- typu int, pozycja na ktorej jest umieszczana wartosc.

4.14.3.4 void Lista::push (int *wartosc*) [inline],[virtual]

Przeciazenie operacji push. Umieszcza element domyslnie na pozycji 1. Nastepuje inkrementacja rozmiar listy.

Parameters

<i>wartosc</i>	- typu int, wartosc umieszczana na liscie.
----------------	--

Implements [Zasobnik< int >](#).

4.14.3.5 int Lista::size () [virtual]

Metoda zwracajaca wielkosc listy.

Returns

rozmiar - typu int,rozmiar listy.

Implements [Zasobnik< int >](#).

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

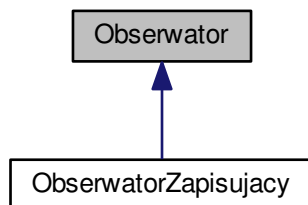
- [Lista.hh](#)
- [Lista.cpp](#)

4.15 Obserwator Class Reference

Klasa [Obserwator](#).

```
#include <Obserwator.hh>
```

Inheritance diagram for Obserwator:



Public Member Functions

- virtual void [odswiez](#) (int k, long int sredni_czas)=0
Metoda odswiezajaca obserwatora.

4.15.1 Detailed Description

Klasa [Obserwator](#).

4.15.2 Member Function Documentation

4.15.2.1 `virtual void Obserwator::odswiez (int k, long int sredni_czas)` `[pure virtual]`

Metoda odswiezajaca obserwatora.

Parameters

<i>k</i>	-typu int, ilosc danych na obserwowany obiekcie
<i>sredni_czas</i>	- typu long int, sredni czas wykonania operacji

Implemented in [ObserwatorZapisujacy](#).

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

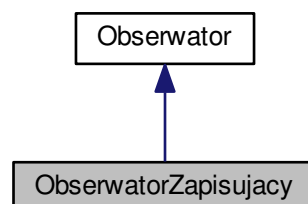
- [Obserwator.hh](#)

4.16 ObserwatorZapisujacy Class Reference

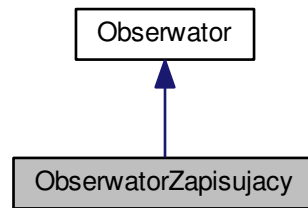
Klasa [ObserwatorZapisujacy](#).

```
#include <ObserwatorZapisujacy.hh>
```

Inheritance diagram for ObserwatorZapisujacy:



Collaboration diagram for ObserwatorZapisujacy:



Public Member Functions

- void `odswiez` (int, long int)

Metoda odswiezajaca obserwatora.

4.16.1 Detailed Description

Klasa `ObserwatorZapisujacy`.

4.16.2 Member Function Documentation

4.16.2.1 void ObserwatorZapisujacy::odswiez (int *k*, long *sredni_czas*) [virtual]

Metoda odswiezajaca obserwatora.

Parameters

<i>k</i>	-typu int, ilosc danych na obserwowany obiekcie
<i>sredni_czas</i>	- typu long int, sredni czas wykonania operacji

Implements `Obserwator`.

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

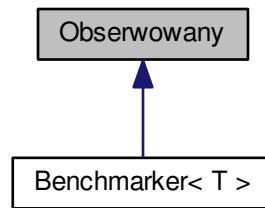
- `ObserwatorZapisujacy.hh`
- `ObserwatorZapisujacy.cpp`

4.17 Obserwowany Class Reference

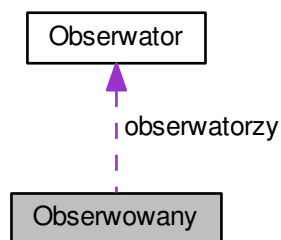
Szablon klasy `Obserwowany`.

```
#include <Obserwowany.hh>
```

Inheritance diagram for Obserwowany:



Collaboration diagram for Obserwowany:



Public Member Functions

- void `dodaj` (`Obserwator` *Obs)
Metoda dodajaca obserwatora do obiektu.
- void `usun` (`Obserwator` *Obs)
Metoda usuwajaca obserwatora z obiektu.

Protected Attributes

- `Obserwator` * `obserwatorzy`

4.17.1 Detailed Description

Szablon klasy `Obserwowany`.

4.17.2 Member Function Documentation

4.17.2.1 void Obserwowany::dodaj(Obserwator * *Obs*) [inline]

Metoda dodająca obserwatora do obiektu.

Parameters

<i>Obs</i>	- typu Obserwator*, wskaznik na danego obserwatora
------------	--

4.17.2.2 void Obserwowany::usun (Obserwator * *Obs*) [inline]

Metoda usuwajaca obserwatora z obiektu.

Parameters

<i>Obs</i>	- typu Obserwator*, wskaznik na danego obserwatora
------------	--

4.17.3 Member Data Documentation

4.17.3.1 Obserwator* Obserwowany::obserwatorzy [protected]

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

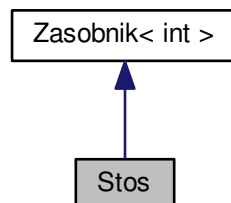
- [Obserwowany.hh](#)

4.18 Stos Class Reference

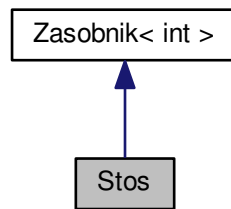
Klasa [Stos](#).

```
#include <Stos.hh>
```

Inheritance diagram for Stos:



Collaboration diagram for Stos:



Public Member Functions

- [Stos](#) ()
Konstruktor bezparametryczny. Konstruktor inicjalizujący strażnika stosu wartością NULL ,oraz rozmiar wartością 0.
- [~Stos](#) ()
Destruktor bezparametryczny stosu.
- void [push](#) (int)
Metoda umieszczająca element na stosie Metoda inkrementuje rozmiar podczas umieszczania elementu na stosie.
- int [pop](#) ()
Metoda zdejmująca element ze stosu. Metoda dekrementuje rozmiar przy zdejmowaniu ze stosu.
- int [pop](#) (int a)
Metoda zdejmująca element ze stosu. Metoda dekrementuje rozmiar przy zdejmowaniu ze stosu.
- int [size](#) ()
Metoda zwracająca wielkość stosu.

4.18.1 Detailed Description

Klasa [Stos](#).

4.18.2 Constructor & Destructor Documentation

4.18.2.1 Stos::Stos ()

Konstruktor bezparametryczny. Konstruktor inicjalizujący strażnika stosu wartością NULL ,oraz rozmiar wartością 0.

4.18.2.2 Stos::~~Stos ()

Destruktor bezparametryczny stosu.

4.18.3 Member Function Documentation

4.18.3.1 int Stos::pop () [virtual]

Metoda zdejmująca element ze stosu. Metoda dekrementuje rozmiar przy zdejmowaniu ze stosu.

Returns

wartosc - typu int, wartosc zdejmowana ze stosu.

Implements [Zasobnik< int >](#).

4.18.3.2 int Stos::pop (int *a*) [inline],[virtual]

Metoda zdejmująca element ze stosu. Metoda dekrementuje rozmiar przy zdejmowaniu ze stosu.

Returns

wartosc - typu int, wartosc zdejmowana ze stosu.

Implements [Zasobnik< int >](#).

4.18.3.3 void Stos::push (int *wartosc*) [virtual]

Metoda umieszczająca element na stosie Metoda inkrementuje rozmiar podczas umieszczania elementu na stosie.

Parameters

<i>wartosc</i>	- typu int, wartosc umieszczana na stosie.
----------------	--

Implements [Zasobnik< int >](#).

4.18.3.4 int Stos::size () [virtual]

Metoda zwracająca wielkość stosu.

Returns

rozmiar - typu int, rozmiar stosu.

Implements [Zasobnik< int >](#).

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

- [Stos.hh](#)
- [Stos.cpp](#)

4.19 Zasobnik< T > Class Template Reference

Szablon klasy [Zasobnik](#).

```
#include <Zasobnik.hh>
```

Public Member Functions

- virtual [~Zasobnik](#) ()
Destruktor wirtualny.
- virtual void [push](#) (T wartosc)=0
Metoda umieszczająca element na zasobniku.
- virtual T [pop](#) ()=0
Metoda zdejmująca element z zasobnika.
- virtual T [pop](#) (T szukana)=0

Metoda zdejmujaca dany element z zasobnika.

- virtual int [size](#) ()=0

Metoda zwracajaca rozmiar zasobnika.

4.19.1 Detailed Description

template<typename T>class Zasobnik< T >

Szablon klasy [Zasobnik](#).

4.19.2 Constructor & Destructor Documentation

4.19.2.1 template<typename T> virtual Zasobnik< T >::~~Zasobnik () [inline],[virtual]

Destruktor wirtualny.

4.19.3 Member Function Documentation

4.19.3.1 template<typename T> virtual T Zasobnik< T >::pop () [pure virtual]

Metoda zdejmujaca element z zasobnika.

Returns

wartosc - typu T, wartosc zdejmowana z zasobnika.

Implemented in [HaszTab](#), [Lista](#), [ArrayLista](#), [DrzewoRB](#), [DrzewoBinarne](#), [Kolejka](#), and [Stos](#).

4.19.3.2 template<typename T> virtual T Zasobnik< T >::pop (T *szukana*) [pure virtual]

Metoda zdejmujaca dany element z zasobnika.

Parameters

<i>szukana</i>	- typu T, wartosc szukana w zasobniku.
----------------	--

Returns

wartosc - typu T, wartosc zdejmowana z zasobnika.

Implemented in [DrzewoRB](#), [HaszTab](#), [DrzewoBinarne](#), [Kolejka](#), [Stos](#), and [Lista](#).

4.19.3.3 template<typename T> virtual void Zasobnik< T >::push (T *wartosc*) [pure virtual]

Metoda umieszczajaca element na zasobniku.

Parameters

<i>wartosc</i>	- typu T, wartosc umieszczana na zasobniku.
----------------	---

Implemented in [HaszTab](#), [Lista](#), [ArrayLista](#), [DrzewoRB](#), [DrzewoBinarne](#), [Kolejka](#), and [Stos](#).

4.19.3.4 `template<typename T> virtual int Zasobnik<T>::size () [pure virtual]`

Metoda zwracajaca rozmiar zasobnika.

Returns

rozmiar - typu int, rozmiar zasobnika.

Implemented in [HaszTab](#), [Lista](#), [ArrayLista](#), [DrzewoRB](#), [DrzewoBinarne](#), [Kolejka](#), and [Stos](#).

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

- [Zasobnik.hh](#)

Chapter 5

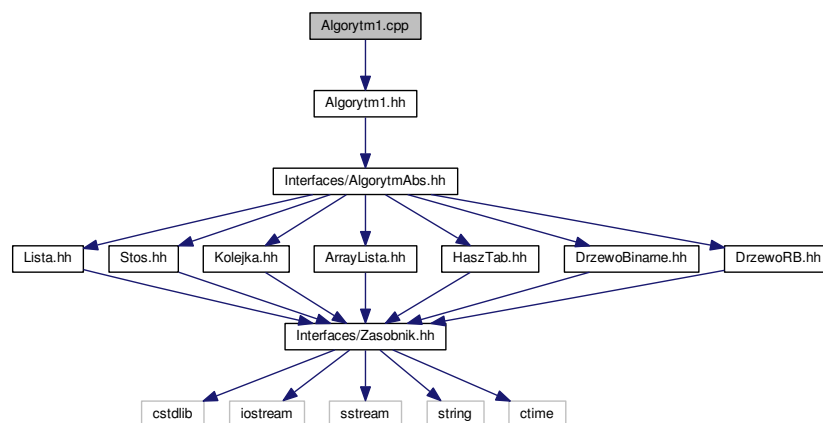
File Documentation

5.1 Algorytm1.cpp File Reference

Metody klasy [Algorytm1](#).

```
#include "Algorytm1.hh"
```

Include dependency graph for Algorytm1.cpp:



5.1.1 Detailed Description

Metody klasy [Algorytm1](#).

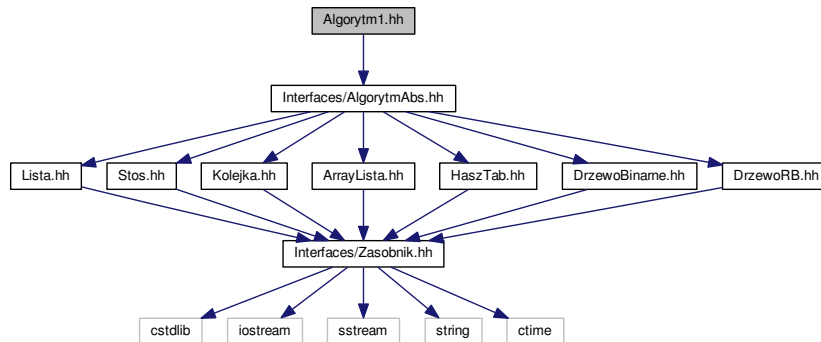
Plik zawiera metody klasy [Algorytm1](#).

5.2 Algorytm1.hh File Reference

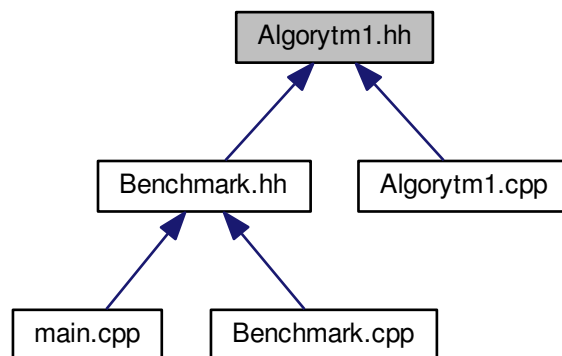
Definicja klasy [Algorytm1](#).

```
#include "Interfaces/AlgorytmAbs.hh"
```

Include dependency graph for Algorytm1.hh:



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



Classes

- class [Algorytm1](#)
Klasa [Algorytm1](#).

5.2.1 Detailed Description

Definicja klasy [Algorytm1](#).

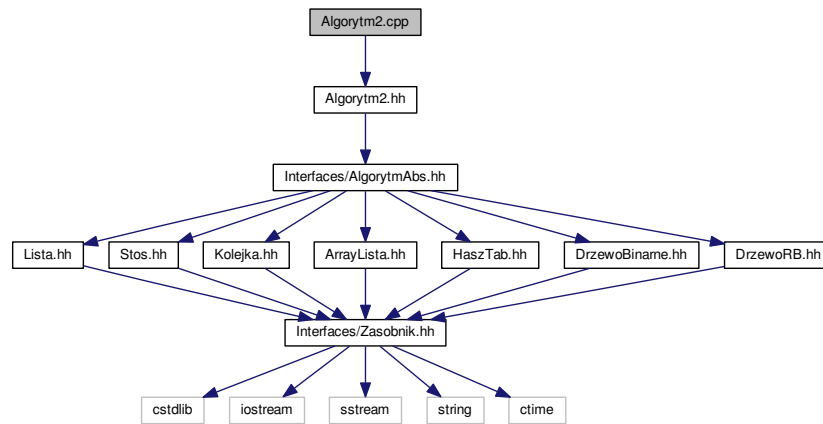
Plik zawiera definicje klasy [Algorytm1](#).

5.3 Algorytm2.cpp File Reference

Metody klasy [Algorytm2](#).

```
#include "Algorytm2.hh"
```

Include dependency graph for Algorytm2.cpp:



5.3.1 Detailed Description

Metody klasy [Algorytm2](#).

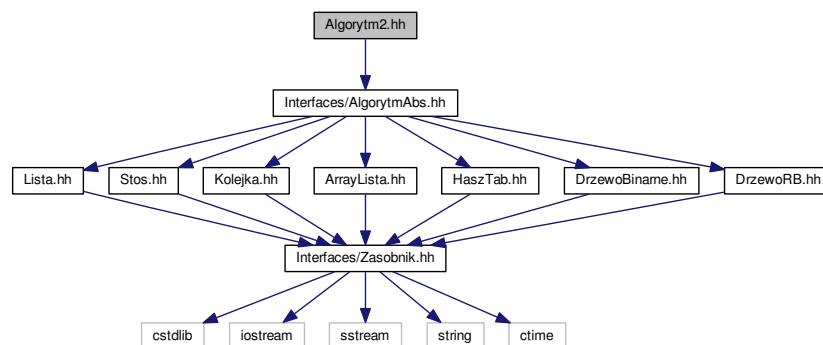
Plik zawiera metody klasy [Algorytm2](#).

5.4 Algorytm2.hh File Reference

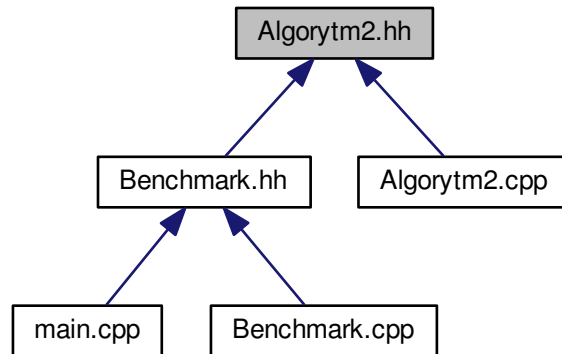
Definicja klasy [Algorytm2](#).

```
#include "Interfaces/AlgorytmAbs.hh"
```

Include dependency graph for Algorytm2.hh:



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



Classes

- class [Algorytm2](#)

Klasa [Algorytm2](#).

5.4.1 Detailed Description

Definicja klasy [Algorytm2](#).

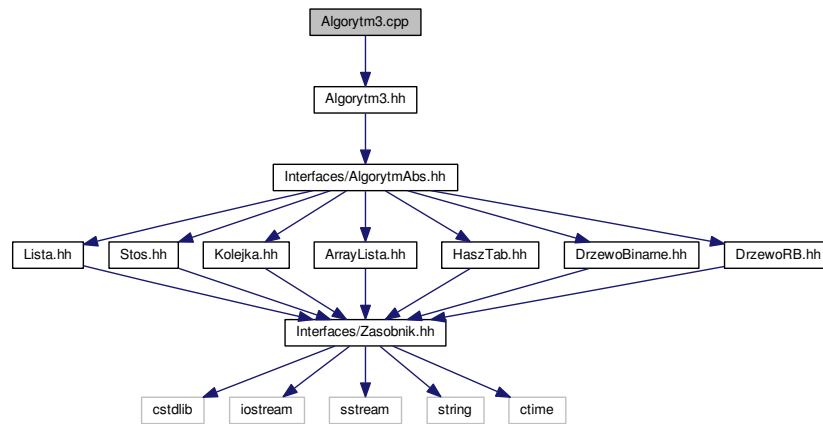
Plik zawiera definicje klasy [Algorytm2](#).

5.5 Algorytm3.cpp File Reference

Metody klasy [Algorytm3](#).

```
#include "Algorytm3.hh"
```

Include dependency graph for Algorytm3.cpp:



5.5.1 Detailed Description

Metody klasy [Algorytm3](#).

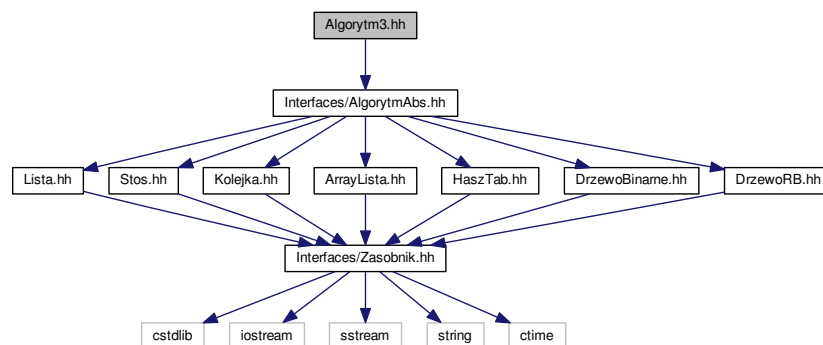
Plik zawiera metody klasy [Algorytm3](#).

5.6 Algorytm3.hh File Reference

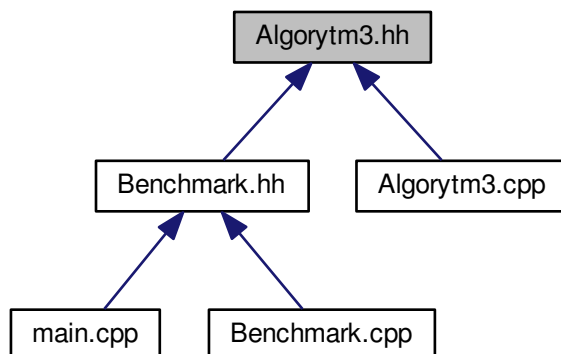
Definicja klasy [Algorytm3](#).

```
#include "Interfaces/AlgorytmAbs.hh"
```

Include dependency graph for Algorytm3.hh:



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



Classes

- class [Algorytm3](#)

Klasa [Algorytm3](#).

5.6.1 Detailed Description

Definicja klasy [Algorytm3](#).

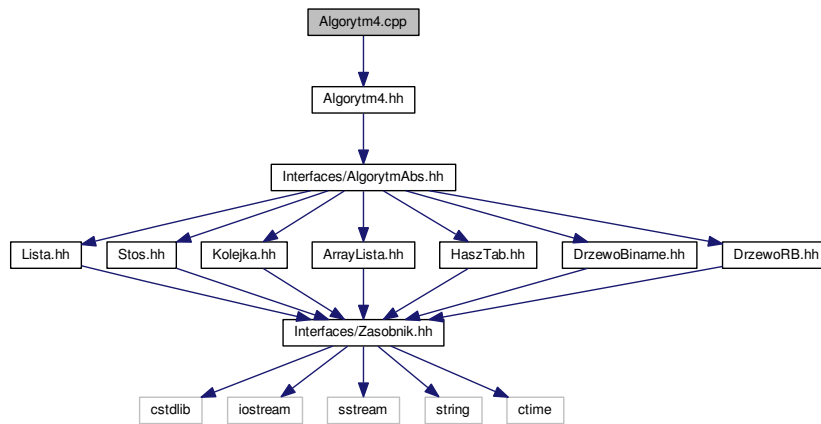
Plik zawiera definicje klasy [Algorytm3](#).

5.7 Algorytm4.cpp File Reference

Metody klasy [Algorytm4](#).

```
#include "Algorytm4.hh"
```

Include dependency graph for Algorytm4.cpp:



5.7.1 Detailed Description

Metody klasy [Algorytm4](#).

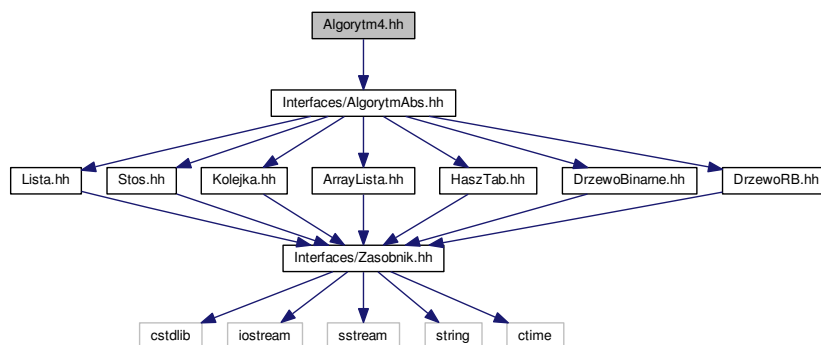
Plik zawiera metody klasy [Algorytm4](#).

5.8 Algorytm4.hh File Reference

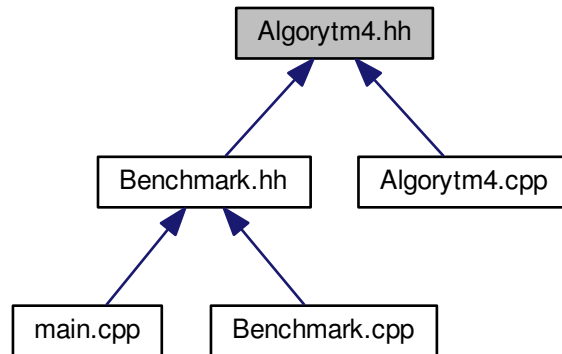
Definicja klasy [Algorytm4](#).

```
#include "Interfaces/AlgorytmAbs.hh"
```

Include dependency graph for Algorytm4.hh:



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



Classes

- class [Algorytm4](#)

Klasa [Algorytm4](#).

5.8.1 Detailed Description

Definicja klasy [Algorytm4](#).

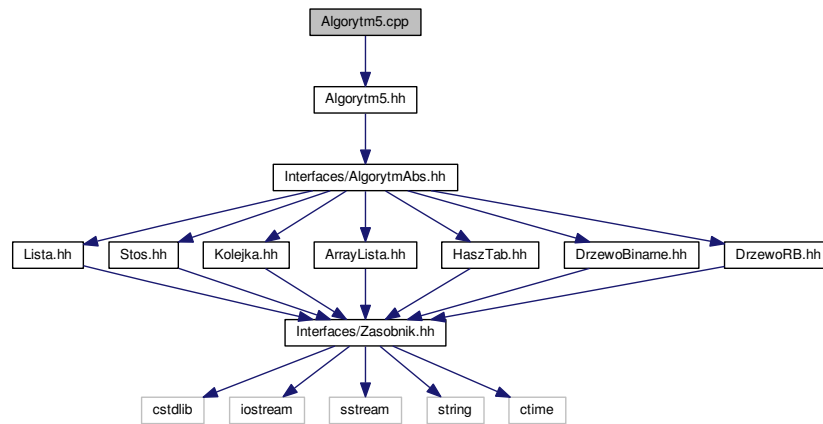
Plik zawiera definicje klasy [Algorytm4](#).

5.9 Algorytm5.cpp File Reference

Metody klasy [Algorytm5](#).


```
#include "Algorytm5.hh"
```

Include dependency graph for Algorytm5.cpp:



5.9.1 Detailed Description

Metody klasy [Algorytm5](#).

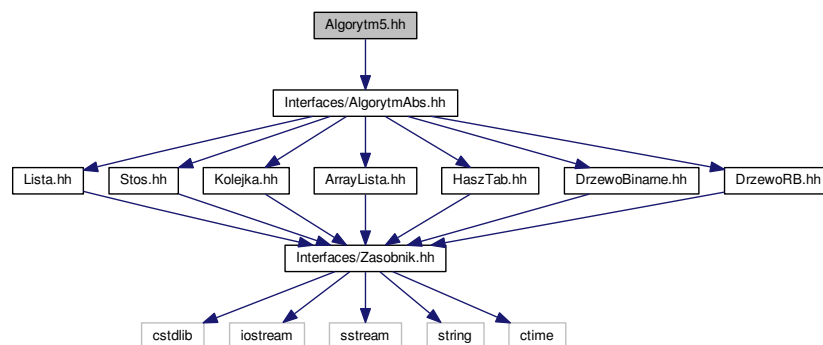
Plik zawiera metody klasy [Algorytm5](#).

5.10 Algorytm5.hh File Reference

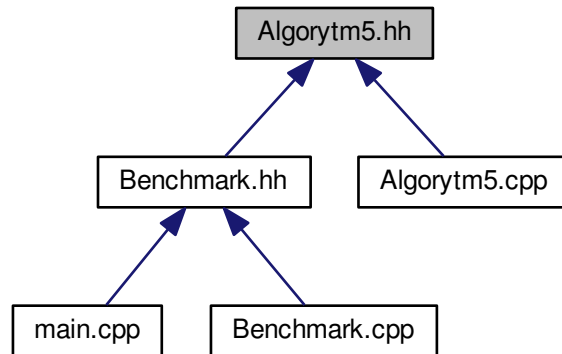
Definicja klasy [Algorytm5](#).

```
#include "Interfaces/AlgorytmAbs.hh"
```

Include dependency graph for Algorytm5.hh:



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



Classes

- class [Algorytm5](#)

Klasa [Algorytm5](#).

5.10.1 Detailed Description

Definicja klasy [Algorytm5](#).

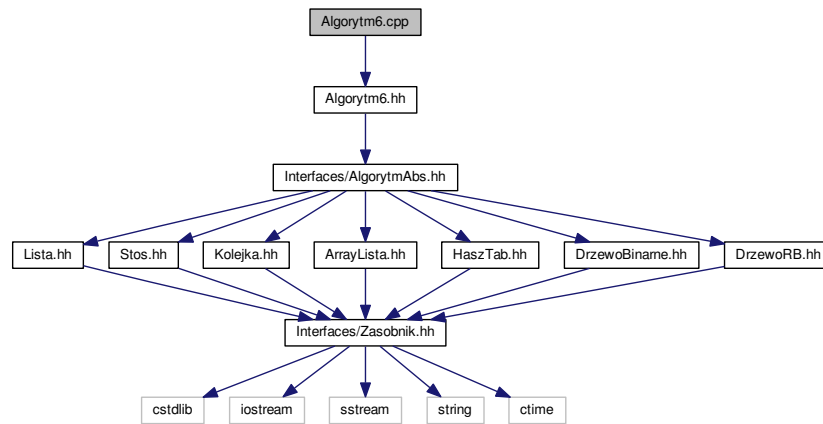
Plik zawiera definicje klasy [Algorytm5](#).

5.11 Algorytm6.cpp File Reference

Metody klasy [Algorytm6](#).

```
#include "Algorytm6.hh"
```

Include dependency graph for Algorytm6.cpp:



5.11.1 Detailed Description

Metody klasy [Algorytm6](#).

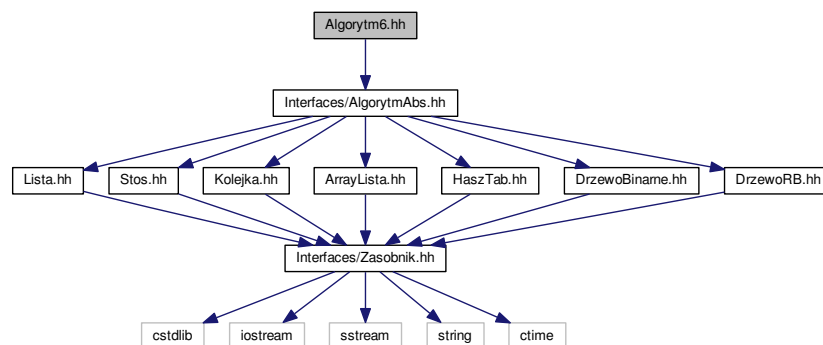
Plik zawiera metody klasy [Algorytm6](#).

5.12 Algorytm6.hh File Reference

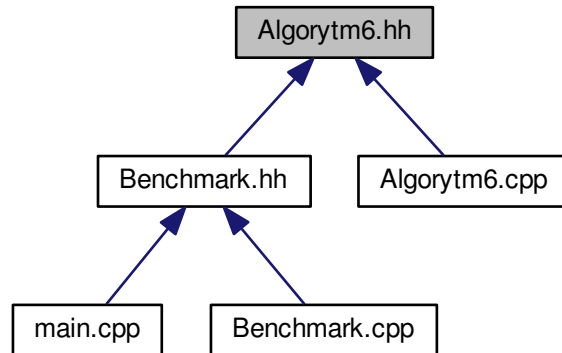
Definicja klasy [Algorytm6](#).

```
#include "Interfaces/AlgorytmAbs.hh"
```

Include dependency graph for Algorytm6.hh:



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



Classes

- class [Algorytm6](#)

Klasa [Algorytm6](#).

5.12.1 Detailed Description

Definicja klasy [Algorytm6](#).

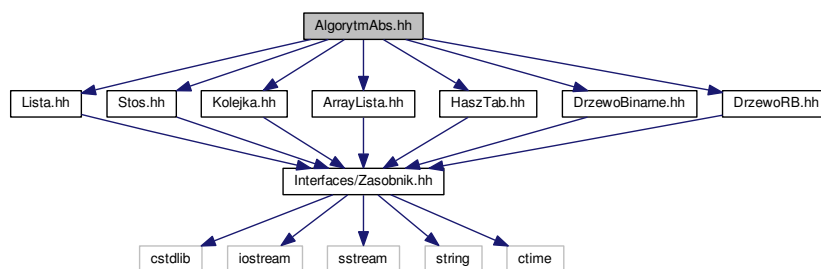
Plik zawiera definicje klasy [Algorytm6](#).

5.13 AlgorytmAbs.hh File Reference

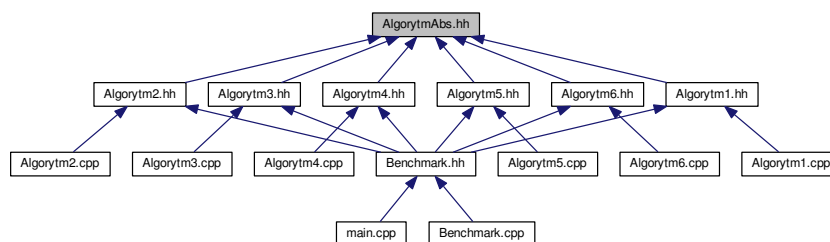
Definicja szablonu klasy abstrakcyjnej [Algorytm](#).

```
#include "Lista.hh"
#include "Stos.hh"
#include "Kolejka.hh"
#include "ArrayLista.hh"
#include "HaszTab.hh"
#include "DrzewoBinarne.hh"
#include "DrzewoRB.hh"
```

Include dependency graph for AlgorytmAbs.hh:



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



Classes

- class [Algorytm< T >](#)

Szablon klasy [Algorytm](#).

5.13.1 Detailed Description

Definicja szablonu klasy abstrakcyjnej [Algorytm](#).

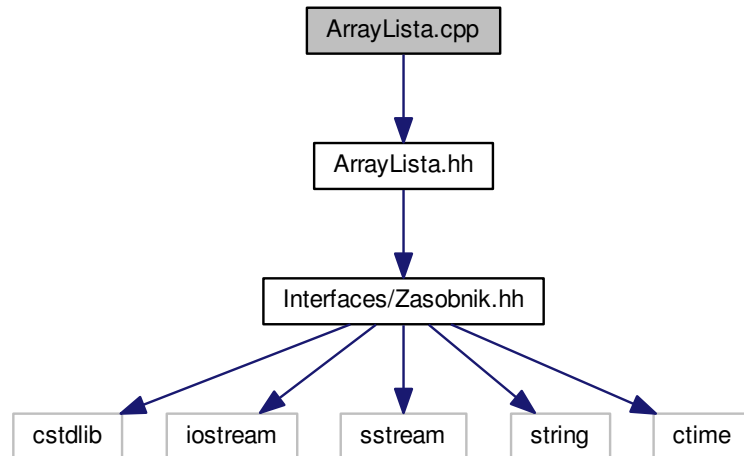
Plik zawiera definicje szablonu klasy abstrakcyjnej [Algorytm](#).

5.14 ArrayLista.cpp File Reference

Metody klasy [ArrayLista](#).

```
#include "ArrayLista.hh"
```

Include dependency graph for ArrayLista.cpp:



5.14.1 Detailed Description

Metody klasy [ArrayLista](#).

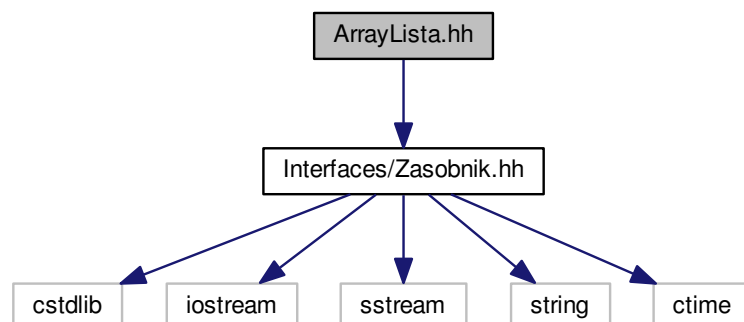
Plik zawiera metody klasy [ArrayLista](#).

5.15 ArrayLista.hh File Reference

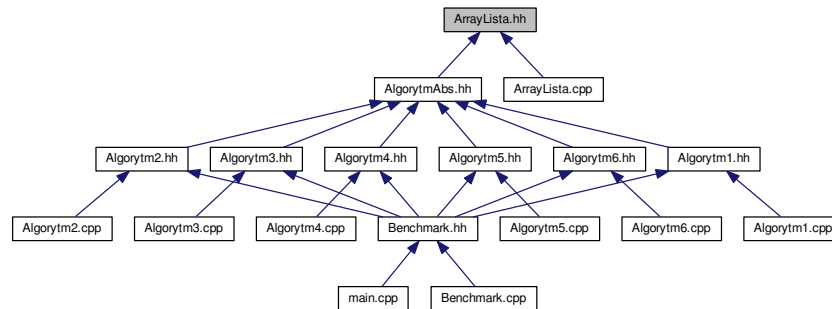
Definicja klasy [ArrayLista](#).

```
#include "Interfaces/Zasobnik.hh"
```

Include dependency graph for ArrayLista.hh:



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



Classes

- class [ArrayLista](#)
Klasa [ArrayLista](#).

5.15.1 Detailed Description

Definicja klasy [ArrayLista](#).

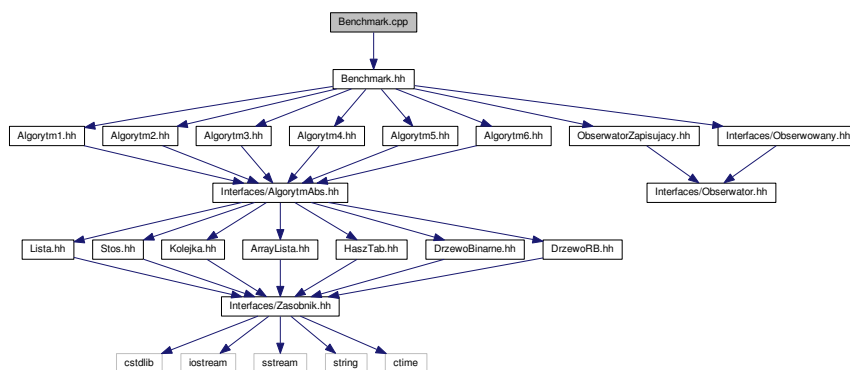
Plik zawiera definicje klasy modulujacej pojecie listy jednokierunkowej opartej na tablicy dynamicznej.

5.16 Benchmark.cpp File Reference

Metody klasy [Benchmark](#).

```
#include "Benchmark.hh"
```

Include dependency graph for Benchmark.cpp:



5.16.1 Detailed Description

Metody klasy [Benchmark](#).

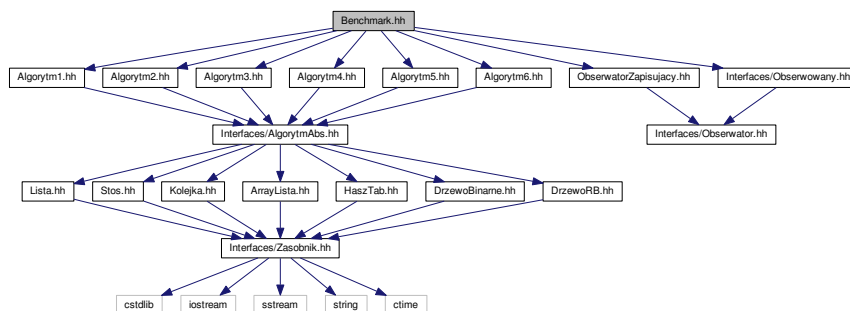
Plik zawiera metody klasy [Benchmark](#).

5.17 Benchmark.hh File Reference

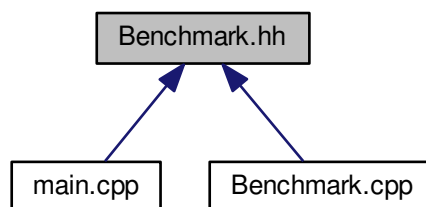
Definicja szablonu klasy [Benchmarker](#).

```
#include "Algorytm1.hh"
#include "Algorytm2.hh"
#include "Algorytm3.hh"
#include "Algorytm4.hh"
#include "Algorytm5.hh"
#include "Algorytm6.hh"
#include "ObserwatorZapisujacy.hh"
#include "Interfaces/Obserwowany.hh"
```

Include dependency graph for Benchmark.hh:



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



Classes

- class [Benchmarker](#)< T >
Szablon klasy [Benchmarker](#).

5.17.1 Detailed Description

Definicja szablonu klasy [Benchmarker](#).

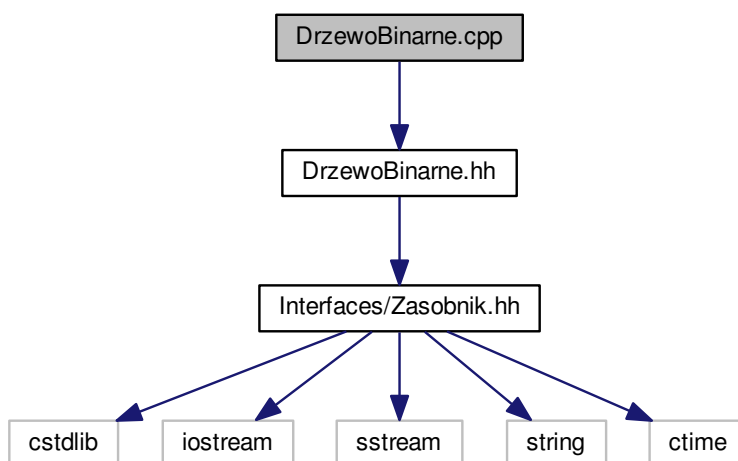
Plik zawiera definicje szablonu klasy [Benchmarker](#).

5.18 DrzewoBinarne.cpp File Reference

Metody klasy [DrzewoBinarne](#).

```
#include "DrzewoBinarne.hh"
```

Include dependency graph for DrzewoBinarne.cpp:



5.18.1 Detailed Description

Metody klasy [DrzewoBinarne](#).

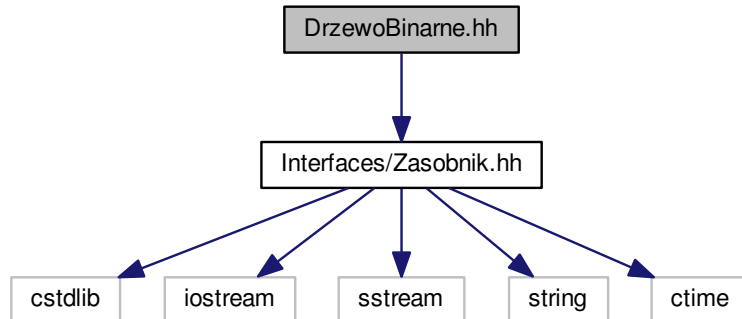
Plik zawiera metody klasy [DrzewoBinarne](#).

5.19 DrzewoBinarne.hh File Reference

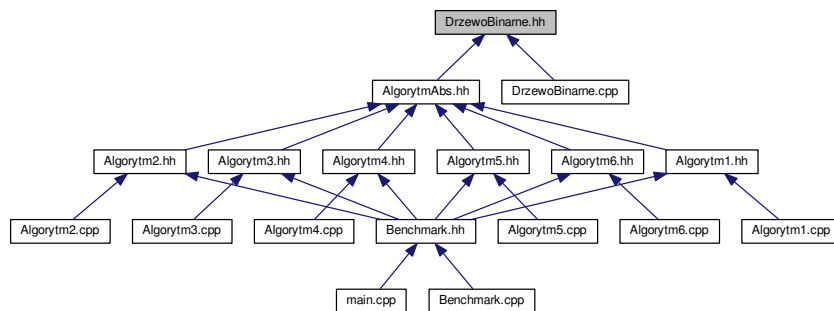
Definicja klasy [DrzewoBinarne](#).

```
#include "Interfaces/Zasobnik.hh"
```

Include dependency graph for DrzewoBinarne.hh:



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



Classes

- class [DrzewoBinarne](#)
Klasa [DrzewoBinarne](#).

5.19.1 Detailed Description

Definicja klasy [DrzewoBinarne](#).

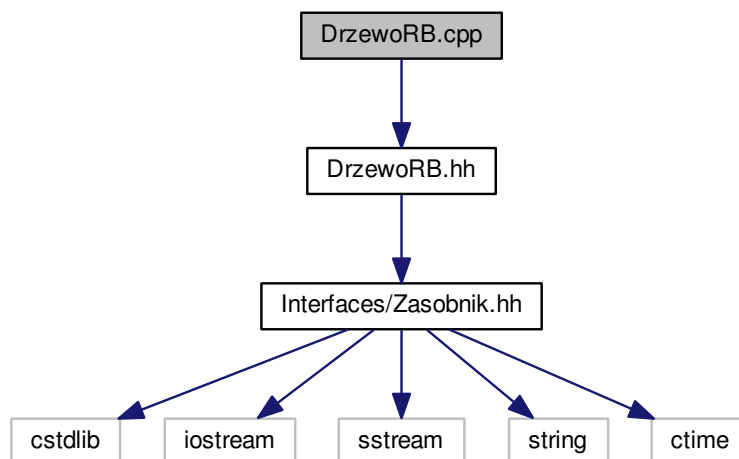
Plik zawiera definicje klasy [DrzewoBinarne](#).

5.20 DrzewoRB.cpp File Reference

Metody klasy [DrzewoRB](#).

```
#include "DrzewoRB.hh"
```

Include dependency graph for DrzewoRB.cpp:



5.20.1 Detailed Description

Metody klasy [DrzewoRB](#).

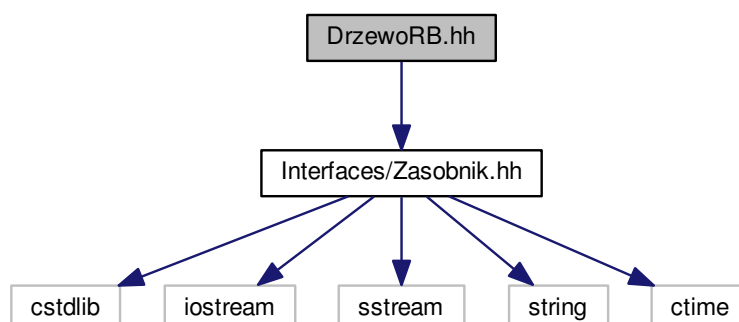
Plik zawiera metody klasy [DrzewoRB](#).

5.21 DrzewoRB.hh File Reference

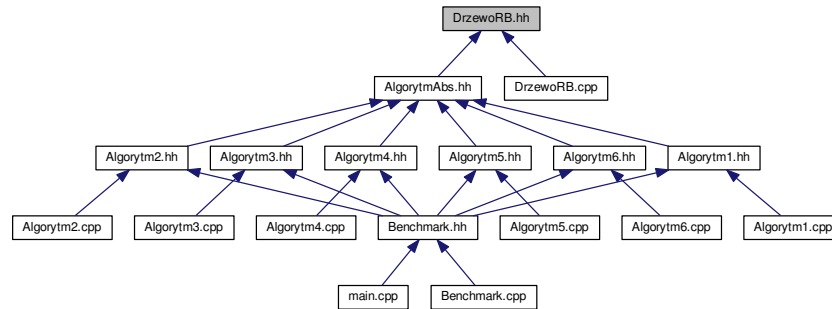
Definicja klasy [DrzewoRB](#).

```
#include "Interfaces/Zasobnik.hh"
```

Include dependency graph for DrzewoRB.hh:



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



Classes

- class [DrzewoRB](#)
Klasa [DrzewoRB](#).

5.21.1 Detailed Description

Definicja klasy [DrzewoRB](#).

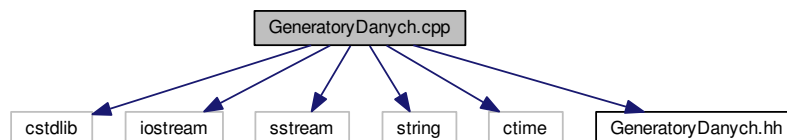
Plik zawiera definicje klasy [DrzewoRB](#).

5.22 GeneratoryDanych.cpp File Reference

Funkcje generacji danych.

```
#include <cstdlib>
#include <iostream>
#include <sstream>
#include <string>
#include <ctime>
#include "GeneratoryDanych.hh"
```

Include dependency graph for GeneratoryDanych.cpp:



Functions

- template<>
int * [generuj dane](#) (int l_danych)

Szablon metody generujacej wartosci losowe danego typu.

- `template<>`
`string * generujdane (int l_danych)`

Szablon metody generujacej wartosci losowe danego typu.

5.22.1 Detailed Description

Funkcje generacji danych.

Plik zawiera funkcje generacji danych .

5.22.2 Function Documentation

5.22.2.1 `template<> int* generujdane (int l_danych)`

Szablon metody generujacej wartosci losowe danego typu.

Parameters

<code>l_danych</code>	- typu int, liczba generowanych danych.
-----------------------	---

Returns

`T*` -wskaznik na dany typ, wskaznik na tablice z wygenerowanymi danymi.

5.22.2.2 `template<> string* generujdane (int l_danych)`

Szablon metody generujacej wartosci losowe danego typu.

Parameters

<code>l_danych</code>	- typu int, liczba generowanych danych.
-----------------------	---

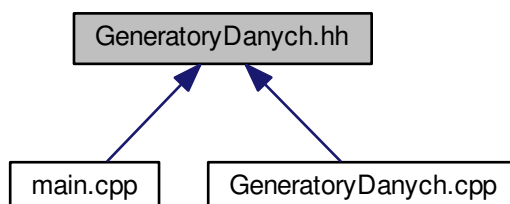
Returns

`T*` -wskaznik na dany typ, wskaznik na tablice z wygenerowanymi danymi.

5.23 GeneratoryDanych.hh File Reference

Szablon funkcji generacji danych.

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



Functions

- `template<typename T>`
`T * generuj dane (int l_danych)`

Szablon metody generujacej wartosci losowe danego typu.

5.23.1 Detailed Description

Szablon funkcji generacji danych.

Plik zawiera szablon funkcji generacji danych.

5.23.2 Function Documentation

5.23.2.1 `template<typename T> T* generuj dane (int l_danych)`

Szablon metody generujacej wartosci losowe danego typu.

Parameters

<code>l_danych</code>	- typu int, liczba generowanych danych.
-----------------------	---

Returns

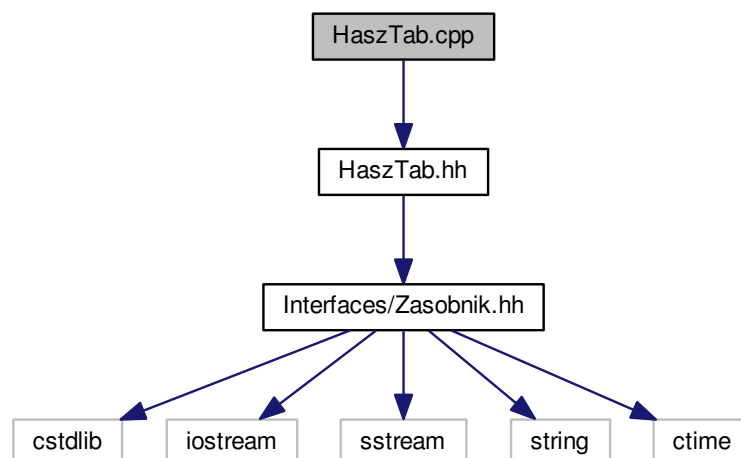
`T*` -wskaznik na dany typ, wskaznik na tablice z wygenerowanymi danymi.

5.24 HaszTab.cpp File Reference

Metody klasy [HaszTab](#).

```
#include "HaszTab.hh"
```

Include dependency graph for HaszTab.cpp:



5.24.1 Detailed Description

Metody klasy [HaszTab](#).

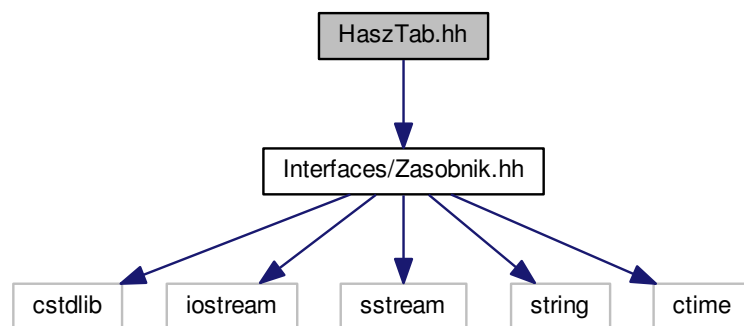
Plik zawiera metody klasy [HaszTab](#).

5.25 HaszTab.hh File Reference

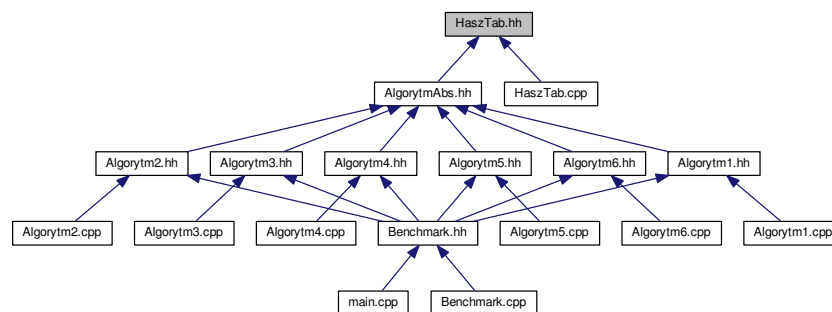
Definicja klasy [HaszTab](#).

```
#include "Interfaces/Zasobnik.hh"
```

Include dependency graph for HaszTab.hh:



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



Classes

- class [HaszTab](#)
Klasa [HaszTab](#).

5.25.1 Detailed Description

Definicja klasy [HaszTab](#).

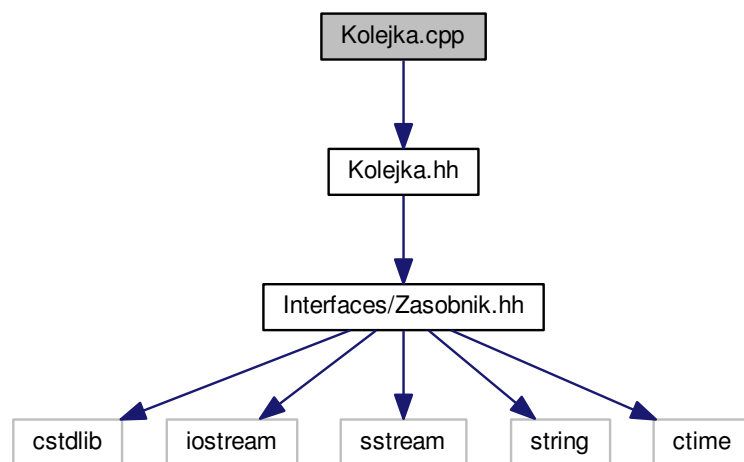
Plik zawiera definicje klasy modulujacej pojecie tablicy haszujacej.

5.26 Kolejka.cpp File Reference

Metody klasy [Kolejka](#).

```
#include "Kolejka.hh"
```

Include dependency graph for Kolejka.cpp:



5.26.1 Detailed Description

Metody klasy [Kolejka](#).

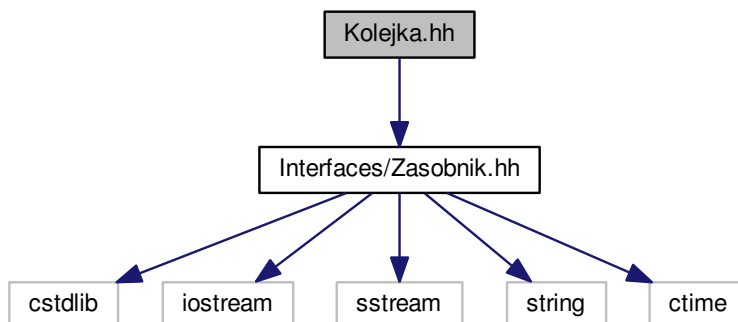
Plik zawiera metody klasy [Kolejka](#).

5.27 Kolejka.hh File Reference

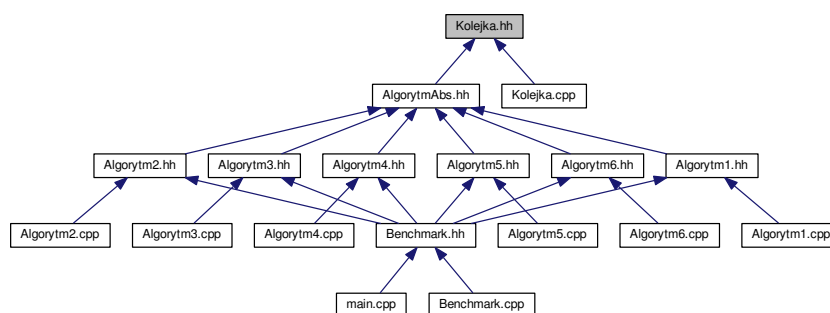
Definicja klasy [Kolejka](#).


```
#include "Interfaces/Zasobnik.hh"
```

Include dependency graph for Kolejka.hh:



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



Classes

- class [Kolejka](#)
Klasa [Kolejka](#).

5.27.1 Detailed Description

Definicja klasy [Kolejka](#).

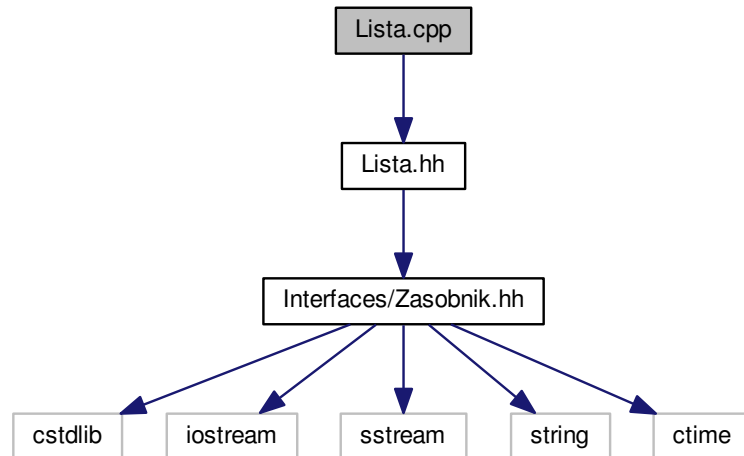
Plik zawiera definicje klasy [Kolejka](#).

5.28 Lista.cpp File Reference

Metody klasy [Lista](#).

```
#include "Lista.hh"
```

Include dependency graph for Lista.cpp:



5.28.1 Detailed Description

Metody klasy [Lista](#).

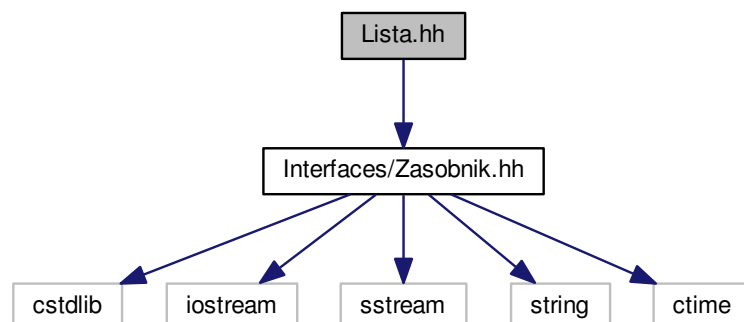
Plik zawiera metody klasy [Lista](#).

5.29 Lista.hh File Reference

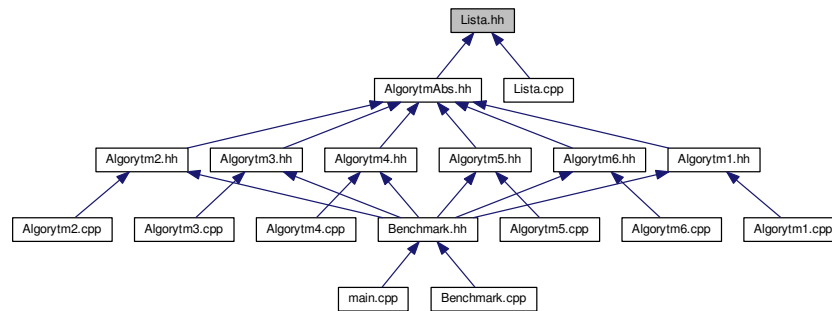
Definicja klasy [Lista](#).

```
#include "Interfaces/Zasobnik.hh"
```

Include dependency graph for Lista.hh:



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



Classes

- class [Lista](#)
Klasa [Lista](#).

5.29.1 Detailed Description

Definicja klasy [Lista](#).

Plik zawiera definicje klasy modulujacej pojecie listy jednokierunkowej.

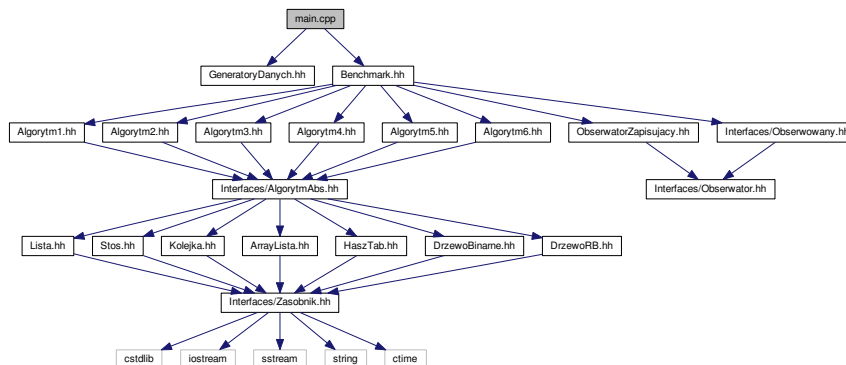
5.30 main.cpp File Reference

Modul glowny.

```
#include "GeneratoryDanych.hh"
```

```
#include "Benchmark.hh"
```

Include dependency graph for main.cpp:



Functions

- int [main](#) (int argc, char *argv[])
Funkcja glowna programu.

5.30.1 Detailed Description

Modul glowny.

Plik zawiera funkcje main.

5.30.2 Function Documentation

5.30.2.1 `int main (int argc, char * argv[])`

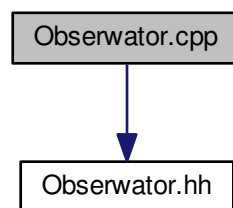
Funkcja glowna programu.

5.31 Obserwator.cpp File Reference

Metody klasy [Obserwator](#).

```
#include "Obserwator.hh"
```

Include dependency graph for Obserwator.cpp:



Functions

- void [odswiez](#) ()

5.31.1 Detailed Description

Metody klasy [Obserwator](#).

Plik zawiera metody klasy [Obserwator](#).

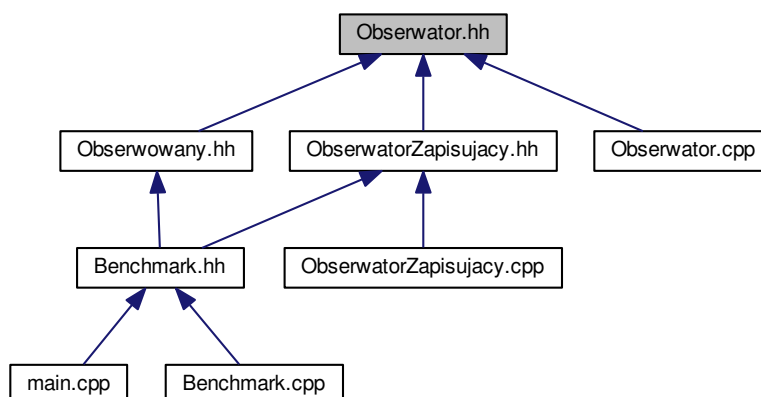
5.31.2 Function Documentation

5.31.2.1 `void odswiez ()`

5.32 Obserwator.hh File Reference

Definicja klasy [Obserwator](#).

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



Classes

- class [Obserwator](#)

Klasa [Obserwator](#).

5.32.1 Detailed Description

Definicja klasy [Obserwator](#).

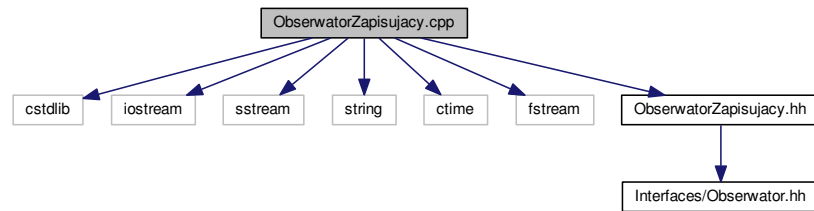
Plik zawiera definicje klasy [Obserwator](#).

5.33 ObserwatorZapisujacy.cpp File Reference

Metody klasy [ObserwatorZapisujacy](#).

```
#include <cstdlib>
#include <iostream>
#include <sstream>
#include <string>
#include <ctime>
#include <fstream>
#include "ObserwatorZapisujacy.hh"
```

Include dependency graph for ObserwatorZapisujacy.cpp:



5.33.1 Detailed Description

Metody klasy [ObserwatorZapisujacy](#).

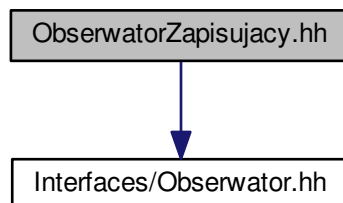
Plik zawiera metody klasy [ObserwatorZapisujacy](#).

5.34 ObserwatorZapisujacy.hh File Reference

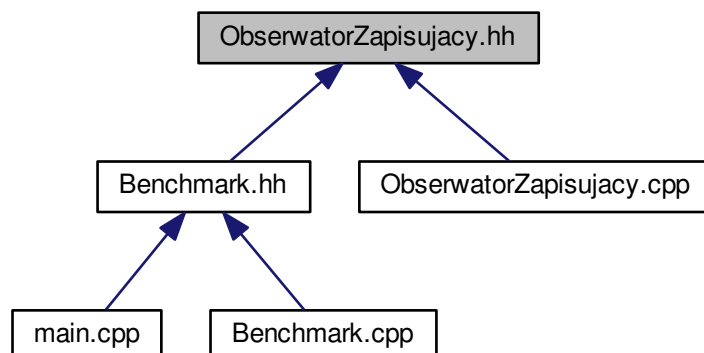
Definicja klasy [ObserwatorZapisujacy](#).

```
#include "Interfaces/Obserwator.hh"
```

Include dependency graph for ObserwatorZapisujacy.hh:



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



Classes

- class [ObserwatorZapisujacy](#)
Klasa [ObserwatorZapisujacy](#).

5.34.1 Detailed Description

Definicja klasy [ObserwatorZapisujacy](#).

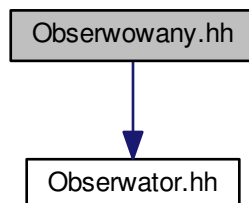
Plik zawiera definicje klasy [ObserwatorZapisujacy](#).

5.35 Obserwowany.hh File Reference

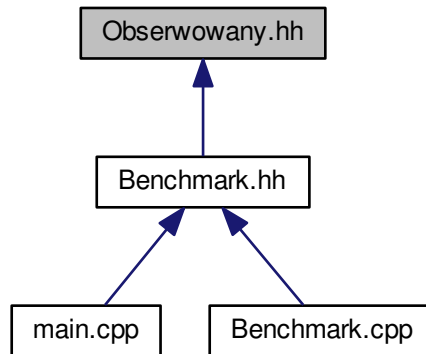
Definicja szablonu klasy abstrakcyjnej [Obserwowany](#).

```
#include "Obserwator.hh"
```

Include dependency graph for `Obserwowany.hh`:



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



Classes

- class [Obserwowany](#)

Szablon klasy [Obserwowany](#).

5.35.1 Detailed Description

Definicja szablonu klasy abstrakcyjnej [Obserwowany](#).

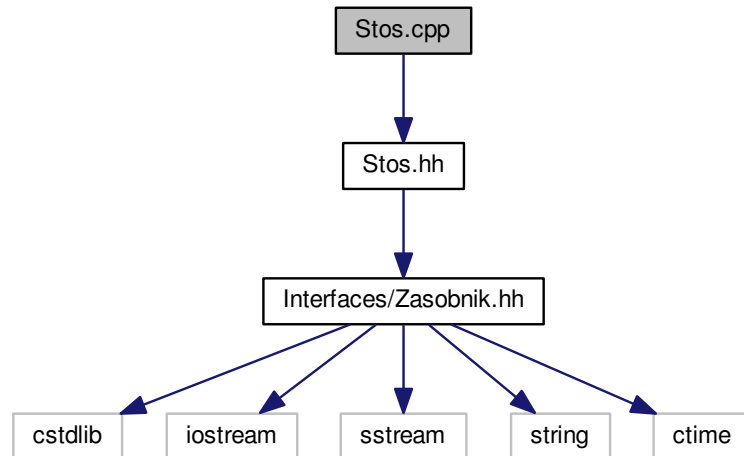
Plik zawiera definicje szablonu klasy abstrakcyjnej [Obserwowany](#).

5.36 Stos.cpp File Reference

Metody klasy [Stos](#).


```
#include "Stos.hh"
```

Include dependency graph for Stos.cpp:



5.36.1 Detailed Description

Metody klasy [Stos](#).

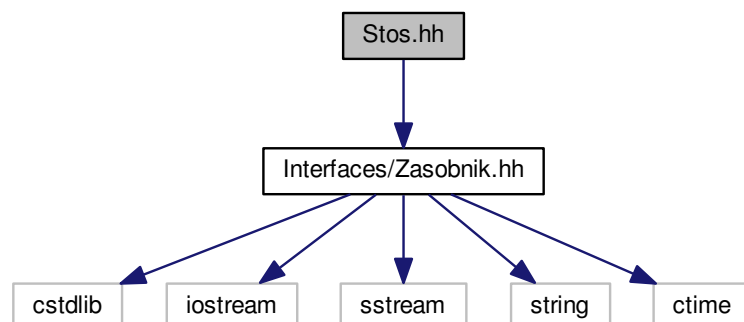
Plik zawiera metody klasy [Stos](#).

5.37 Stos.hh File Reference

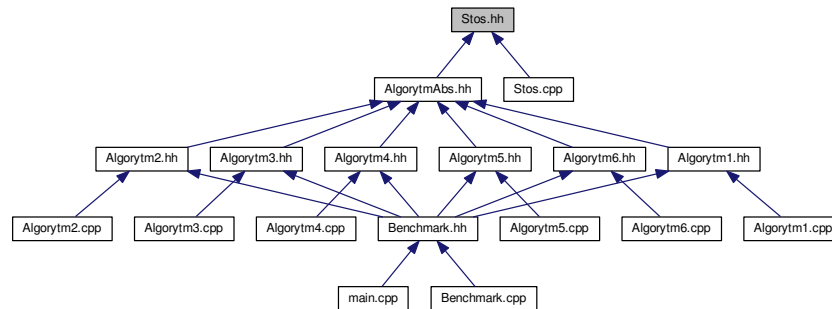
Definicja klasy [Stos](#).

```
#include "Interfaces/Zasobnik.hh"
```

Include dependency graph for Stos.hh:



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



Classes

- class [Stos](#)

Klasa [Stos](#).

5.37.1 Detailed Description

Definicja klasy [Stos](#).

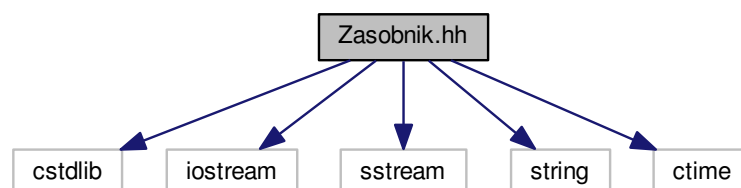
Plik zawiera definicje klasy [Stos](#).

5.38 Zasobnik.hh File Reference

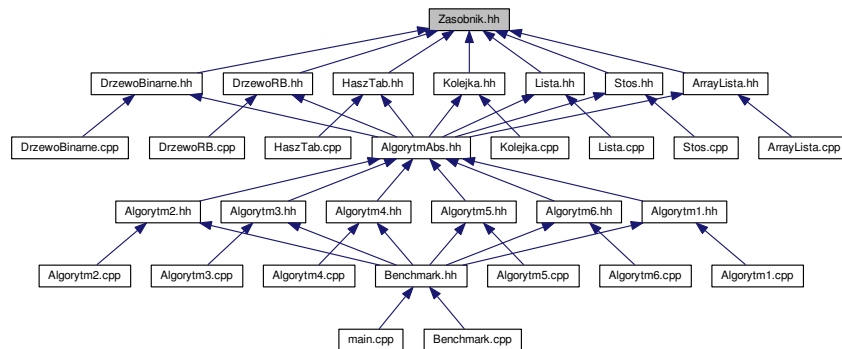
Definicja szablonu klasy abstrakcyjnej [Zasobnik](#).

```
#include <cstdlib>
#include <iostream>
#include <sstream>
#include <string>
#include <ctime>
```

Include dependency graph for Zasobnik.hh:



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



Classes

- class [Zasobnik< T >](#)
Szablon klasy [Zasobnik](#).

5.38.1 Detailed Description

Definicja szablonu klasy abstrakcyjnej [Zasobnik](#).

Plik zawiera definicje szablonu klasy abstrakcyjnej [Zasobnik](#).

Index

- ~Algorytm
 - Algorytm, 7
- ~Algorytm1
 - Algorytm1, 9
- ~Algorytm4
 - Algorytm4, 14
- ~Algorytm6
 - Algorytm6, 21
- ~ArrayLista
 - ArrayLista, 24
- ~DrzewoBinarne
 - DrzewoBinarne, 28
- ~DrzewoRB
 - DrzewoRB, 32
- ~HaszTab
 - HaszTab, 37
- ~Kolejka
 - Kolejka, 41
- ~Lista
 - Lista, 43
- ~Stos
 - Stos, 51
- ~Zasobnik
 - Zasobnik, 53
- Algorytm
 - ~Algorytm, 7
 - alokujdane, 7
 - wykonajalgorytm, 8
- Algorytm < T >, 7
- Algorytm1, 8
 - ~Algorytm1, 9
 - alokujdane, 9
 - wykonajalgorytm, 9
- Algorytm1.cpp, 55
- Algorytm1.hh, 55
- Algorytm2, 10
 - alokujdane, 11
 - wykonajalgorytm, 11
- Algorytm2.cpp, 56
- Algorytm2.hh, 57
- Algorytm3, 11
 - alokujdane, 12
 - wykonajalgorytm, 13
- Algorytm3.cpp, 58
- Algorytm3.hh, 59
- Algorytm4, 13
 - ~Algorytm4, 14
 - alokujdane, 14
 - mergesort, 16
 - scal, 16
 - wykonajalgorytm, 16
- Algorytm4.cpp, 60
- Algorytm4.hh, 61
- Algorytm5, 17
 - alokujdane, 18
 - wykonajalgorytm, 18
- Algorytm5.cpp, 62
- Algorytm5.hh, 63
- Algorytm6, 20
 - ~Algorytm6, 21
 - alokujdane, 21
 - wykonajalgorytm, 22
- Algorytm6.cpp, 64
- Algorytm6.hh, 65
- AlgorytmAbs.hh, 66
- alokujdane
 - Algorytm, 7
 - Algorytm1, 9
 - Algorytm2, 11
 - Algorytm3, 12
 - Algorytm4, 14
 - Algorytm5, 18
 - Algorytm6, 21
- ArrayLista, 22
 - ~ArrayLista, 24
 - ArrayLista, 24
 - pop, 24
 - push, 24
 - size, 24
- ArrayLista.cpp, 67
- ArrayLista.hh, 68
- Benchmark.cpp, 69
- Benchmark.hh, 70
- Benchmarkmarker
 - powiadom, 26
 - testuj, 26
- Benchmarkmarker < T >, 25
- dodaj
 - Obserwowany, 48
- DrzewoBinarne, 27
 - ~DrzewoBinarne, 28
 - DrzewoBinarne, 28
 - obroc_l, 28
 - obroc_p, 29
 - pop, 29
 - push, 29
 - rownowaz, 30

- size, 30
- wypisz, 30
- wypisz_pelne, 31
- DrzewoBinarne.cpp, 71
- DrzewoBinarne.hh, 71
- DrzewoRB, 31
 - ~DrzewoRB, 32
 - DrzewoRB, 32
 - obroc_l, 33
 - obroc_p, 33
 - pop, 33
 - push, 34
 - size, 34
 - ukladaj, 34
 - wypisz, 35
 - wypisz_pelne, 35
- DrzewoRB.cpp, 72
- DrzewoRB.hh, 73
- GeneratoryDanych.cpp, 74
 - generujdane, 75
- GeneratoryDanych.hh, 75
 - generujdane, 76
- generujdane
 - GeneratoryDanych.cpp, 75
 - GeneratoryDanych.hh, 76
- HaszTab, 35
 - ~HaszTab, 37
 - HaszTab, 37
 - mieszaj, 37
 - odczytaj, 37
 - pop, 38
 - push, 38, 39
 - size, 39
 - size_k1, 39
 - size_k2, 39
- HaszTab.cpp, 76
- HaszTab.hh, 77
- Kolejka, 40
 - ~Kolejka, 41
 - Kolejka, 41
 - pop, 41
 - push, 41
 - size, 41
- Kolejka.cpp, 78
- Kolejka.hh, 78
- Lista, 42
 - ~Lista, 43
 - Lista, 43
 - pop, 43
 - push, 43, 45
 - size, 45
- Lista.cpp, 79
- Lista.hh, 80
- main
 - main.cpp, 82
 - main.cpp, 81
 - main, 82
 - mergesort
 - Algorytm4, 16
 - mieszaj
 - HaszTab, 37
 - obroc_l
 - DrzewoBinarne, 28
 - DrzewoRB, 33
 - obroc_p
 - DrzewoBinarne, 29
 - DrzewoRB, 33
 - Obserwator, 45
 - odswiez, 46
 - Obserwator.cpp, 82
 - odswiez, 82
 - Obserwator.hh, 82
 - ObserwatorZapisujacy, 46
 - odswiez, 47
 - ObserwatorZapisujacy.cpp, 83
 - ObserwatorZapisujacy.hh, 84
 - obserwatorzy
 - Obserwowany, 50
 - Obserwowany, 47
 - dodaj, 48
 - obserwatorzy, 50
 - usun, 50
 - Obserwowany.hh, 85
 - odczytaj
 - HaszTab, 37
 - odswiez
 - Obserwator, 46
 - Obserwator.cpp, 82
 - ObserwatorZapisujacy, 47
 - pop
 - ArrayLista, 24
 - DrzewoBinarne, 29
 - DrzewoRB, 33
 - HaszTab, 38
 - Kolejka, 41
 - Lista, 43
 - Stos, 51, 52
 - Zasobnik, 53
 - powiadom
 - Benchmark, 26
 - push
 - ArrayLista, 24
 - DrzewoBinarne, 29
 - DrzewoRB, 34
 - HaszTab, 38, 39
 - Kolejka, 41
 - Lista, 43, 45
 - Stos, 52
 - Zasobnik, 53
- rownowaz

- DrzewoBinarne, [30](#)
- scal
 - Algorytm4, [16](#)
- size
 - ArrayLista, [24](#)
 - DrzewoBinarne, [30](#)
 - DrzewoRB, [34](#)
 - HaszTab, [39](#)
 - Kolejka, [41](#)
 - Lista, [45](#)
 - Stos, [52](#)
 - Zasobnik, [53](#)
- size_k1
 - HaszTab, [39](#)
- size_k2
 - HaszTab, [39](#)
- Stos, [50](#)
 - ~Stos, [51](#)
 - pop, [51](#), [52](#)
 - push, [52](#)
 - size, [52](#)
 - Stos, [51](#)
- Stos.cpp, [86](#)
- Stos.hh, [87](#)
- testuj
 - Benchmark, [26](#)
- ukladaj
 - DrzewoRB, [34](#)
- usun
 - Obserwowany, [50](#)
- wykonajalgorytm
 - Algorytm, [8](#)
 - Algorytm1, [9](#)
 - Algorytm2, [11](#)
 - Algorytm3, [13](#)
 - Algorytm4, [16](#)
 - Algorytm5, [18](#)
 - Algorytm6, [22](#)
- wypisz
 - DrzewoBinarne, [30](#)
 - DrzewoRB, [35](#)
- wypisz_pelne
 - DrzewoBinarne, [31](#)
 - DrzewoRB, [35](#)
- Zasobnik
 - ~Zasobnik, [53](#)
 - pop, [53](#)
 - push, [53](#)
 - size, [53](#)
- Zasobnik< T >, [52](#)
- Zasobnik.hh, [88](#)