Lista, Stos, Kolejka 0.1

Wygenerowano przez Doxygen 1.8.6

Cz, 7 kwi 2016 12:43:20

Spis treści

1	Inde	ks hiera	archiczny									1
	1.1	Hierard	chia klas .				 	 	 	 	 	 1
2	Inde	ks klas										3
	2.1	Lista kl	las				 	 	 	 	 	 3
3	Inde	ks plikć	ów									5
	3.1	Lista p	lików				 	 	 	 	 	 5
4	Dok	umenta	cja klas									7
	4.1	Dokum	entacja sz	ablonu klasy	/ IList< T	yp > .	 	 	 	 	 	 7
		4.1.1	Opis szcz	egółowy			 	 	 	 	 	 7
	4.2	Dokum	entacja sz	ablonu klasy	/ IQueue<	< Typ >	 	 	 	 	 	 8
		4.2.1	Opis szcz	egółowy			 	 	 	 	 	 8
	4.3	Dokum	entacja sz	ablonu klasy	/ IStack<	Typ >	 	 	 	 	 	 8
		4.3.1	Opis szcz	egółowy			 	 	 	 	 	 9
	4.4	Dokum	entacja sz	ablonu klasy	List< Ty	γ p >	 	 	 	 	 	 9
		4.4.1	Opis szcz	egółowy			 	 	 	 	 	 11
		4.4.2	Dokumen	tacja funkcji	składow	ych	 	 	 	 	 	 11
			4.4.2.1	add			 	 	 	 	 	 11
			4.4.2.2	get			 	 	 	 	 	 11
			4.4.2.3	isEmpty .			 	 	 	 	 	 11
			4.4.2.4	remove			 	 	 	 	 	 12
			4.4.2.5	size			 	 	 	 	 	 13
	4.5	Dokum	entacja sz	ablonu klasy	/ Queue<	< Typ >	 	 	 	 	 	 13
		4.5.1	Opis szcz	egółowy			 	 	 	 	 	 14
		4.5.2	Dokumen	tacja funkcji	składow	ych	 	 	 	 	 	 14
			4.5.2.1	dequeue .			 	 	 	 	 	 14
			4.5.2.2	enqueue .			 	 	 	 	 	 15
			4.5.2.3	front			 	 	 	 	 	 15
			4.5.2.4	isEmpty .			 	 	 	 	 	 15
			4.5.2.5	size			 	 	 	 	 	 15

iv SPIS TREŚCI

	4.6	Dokum	entacja szablonu klasy Stack< Typ >	16
		4.6.1	Opis szczegółowy	17
		4.6.2	Dokumentacja funkcji składowych	17
			4.6.2.1 isEmpty	17
			4.6.2.2 pop	17
			4.6.2.3 push	17
			4.6.2.4 size	18
			4.6.2.5 top	18
	4.7	Dokum	entacja klasy Stoper	18
		4.7.1	Opis szczegółowy	19
5	Doku	umenta	cja plików	21
	5.1	Dokum	entacja pliku IList.hh	21
		5.1.1	Opis szczegółowy	22
	5.2	Dokum	entacja pliku lQueue.hh	22
		5.2.1	Opis szczegółowy	23
	5.3	Dokum	entacja pliku IStack.hh	23
		5.3.1	Opis szczegółowy	23
	5.4	Dokum	entacja pliku List.hh	24
		5.4.1	Opis szczegółowy	24
	5.5	Dokum	entacja pliku Queue.hh	24
		5.5.1	Opis szczegółowy	25
	5.6	Dokum	entacja pliku Stack.hh	25
Inc	deks			27

27

Rozdział 1

Indeks hierarchiczny

1.1 Hierarchia klas

Ta lista dziedziczenia posortowana jest z grubsza, choć nie całkowicie, alfabetycznie:

t< Typ $>$	7
$List < Typ > \dots $	9
$neue < Typ > \ldots \ldots \ldots \ldots \ldots$	8
Queue < Typ >	13
$\operatorname{ack}\ldots$	8
Stack< Typ >	16
pper	
Stoper	18

Indeks hierarchiczny

Rozdział 2

Indeks klas

2.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

t< Typ $>$	
Interfejs listy	
eue< Typ >	
Interfejs kolejki	8
ack< Typ >	
Interfejs stosu	8
< Typ >	
Szablon listy	9
eue < Typ >	
Szablon kolejki	13
ck< Typ >	
Szablon stosu	16
per	18

Indeks klas

Rozdział 3

Indeks plików

3.1 Lista plików

Tutaj znajduje się lista wszystkich udokumentowanych plików z ich krótkimi opisami:

IList.hh	
Definicja klasy IList	21
IQueue.hh	
Definicja klasy IQueue	22
IStack.hh	
Definicja klasy IStack	23
List.hh	
Definicja klasy List	24
main.cpp	??
Queue.hh	
Definicja klasy Queue	24
Stack.hh	
Definicja klasy Stack W pliku znajduje sie klasa Stack	25
Stoper.cpp	??
Stoper.hh	??

Indeks plików 6

Rozdział 4

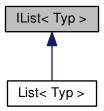
Dokumentacja klas

4.1 Dokumentacja szablonu klasy IList< Typ>

Interfejs listy.

#include <IList.hh>

Diagram dziedziczenia dla IList< Typ>



Metody publiczne

- virtual void add (Typ, int)=0
 - Zapisywanie na liste.
- virtual Typ remove (int)=0

Ściąganie z listy.

• virtual int size ()=0

Rozmiar listy.

• virtual bool isEmpty ()=0

Czy pusty?

virtual Typ get (int)=0

Element listy.

4.1.1 Opis szczegółowy

template<typename Typ>class IList< Typ>

Na liste mozemy wpisywac na każde miejsce i usuwać z każdego miejsca

Definicja w linii 18 pliku IList.hh.

Dokumentacja dla tej klasy została wygenerowana z pliku:

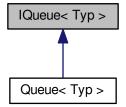
• IList.hh

4.2 Dokumentacja szablonu klasy IQueue < Typ >

Interfejs kolejki.

#include <IQueue.hh>

Diagram dziedziczenia dla IQueue < Typ >



Metody publiczne

- virtual int size ()=0
- virtual bool **isEmpty** ()=0
- virtual void enqueue (Typ)=0
- virtual Typ **dequeue** ()=0
- virtual Typ front ()=0

4.2.1 Opis szczegółowy

template<typename Typ>class IQueue< Typ>

Kolejka jest strukturą typu FIFO (First-In-Fist-Out).

Definicja w linii 17 pliku IQueue.hh.

Dokumentacja dla tej klasy została wygenerowana z pliku:

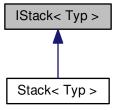
• IQueue.hh

4.3 Dokumentacja szablonu klasy IStack< Typ>

Interfejs stosu.

#include <IStack.hh>

Diagram dziedziczenia dla IStack< Typ >



Metody publiczne

• virtual int size ()=0

Rozmiar stosu.

• virtual bool isEmpty ()=0

Czy pusty?

virtual void push (Typ)=0

Wkładanie na stos.

virtual Typ pop ()=0

Ściąganie ze stosu.

• virtual Typ top ()=0

Szczyt stosu.

4.3.1 Opis szczegółowy

template<typename Typ>class IStack< Typ>

Stos jest strukturą typu LIFO (Last-In-Fist-Out).

Definicja w linii 19 pliku IStack.hh.

Dokumentacja dla tej klasy została wygenerowana z pliku:

· IStack.hh

4.4 Dokumentacja szablonu klasy List< Typ >

Szablon listy.

#include <List.hh>

Diagram dziedziczenia dla List< Typ>

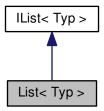
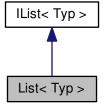


Diagram współpracy dla List< Typ >:



Metody publiczne

- virtual void add (Typ item, int index)
 - Zapisywanie na liste.
- virtual Typ remove (int index) throw (EmptyListException)
 - Ściąganie z listy.
- virtual int size ()
 - Rozmiar listy.
- virtual bool isEmpty ()
 - Czy pusty?
- virtual Typ get (int index)
 - Element listy.

Atrybuty chronione

tablica1D< Typ > Tablica

4.4.1 Opis szczegółowy

template<typename Typ>class List< Typ>

Na liste mozemy wpisywac na każde miejsce i usuwać z każdego miejsca

Definicja w linii 18 pliku List.hh.

4.4.2 Dokumentacja funkcji składowych

```
4.4.2.1 template < typename Typ > virtual void List < Typ >::add ( Typ item, int index ) [inline], [virtual]
```

Wkłada element na dowolne miejsce na liście, jeżeli int index jest większy minimum o 2 od indexu ostatniego elementu to przestrzeń między elementami zostaje uzupełniona przez wartości 0. Jeżeli int index wskazuje na miejsce, gdzie znajduje się element to zostaje on przepisany na kolejną pozycje, a na jego miejsce zostaje wpisany item

Parametry

in	item	- element, który chcemy umieścić na liście
in	index	- miejsce, gdzie chcemy zapisać element (liczony od 0)

Implementuje IList< Typ >.

Definicja w linii 34 pliku List.hh.

4.4.2.2 template<typename Typ > virtual Typ List< Typ >::get(int index) [inline], [virtual]

Dostęp do dowolnego elementu listy

Wyjątki

EmptyListException	wyjątek pustej listy, wyrzucany gdy chcemy odczytać element o indexie którego
	nie ma na liście

Parametry

in	index	- Numer elementu, który chcemy odczytać, gdzie index 0 to pierwszy element

Zwraca

Zwraca element, bez jego usuwania

Implementuje IList< Typ >.

Definicja w linii 84 pliku List.hh.

4.4.2.3 template<typename Typ > virtual bool List< Typ >::isEmpty() [inline], [virtual]

Sprawdza czy na liście znajdują się elementy

Zwracane wartości

true	- lista pusty
false	- na liście są elementy

Implementuje IList< Typ >.

Definicja w linii 70 pliku List.hh.

4.4.2.4 template<typename Typ > virtual Typ List< Typ >::remove(int index) throw EmptyListException) [inline], [virtual]

Usuwa element z listy

Wyjątki

EmptyListException	wyjątek pustej listy, wyrzucany gdy chcemy usunąć element o indexie którego nie	1
	ma na liście	

Zwraca

Zwraca usunięty element

Implementuje IList< Typ >.

Definicja w linii 46 pliku List.hh.

4.4.2.5 template < typename Typ > virtual int List < Typ >::size() [inline], [virtual]

Rozmiar listy jest liczbą całkowitą liczoną od 0, gdy lista pusty,

Zwraca

Zwraca liczbę elementów zapisanych na liście

Implementuje IList< Typ >.

Definicja w linii 61 pliku List.hh.

Dokumentacja dla tej klasy została wygenerowana z pliku:

· List.hh

4.5 Dokumentacja szablonu klasy Queue< Typ>

Szablon kolejki.

#include <Queue.hh>

Diagram dziedziczenia dla Queue < Typ >

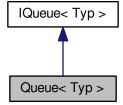
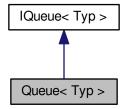


Diagram współpracy dla Queue < Typ >:



Metody publiczne

• virtual int size ()

Rozmiar kolejki.

virtual bool isEmpty ()

Czy pusty?

• virtual void enqueue (Typ item)

Wkładanie do kolejki.

virtual Typ dequeue () throw (EmptyQueueException)

Wychodzenie z kolejki.

• virtual Typ front () throw (EmptyQueueException)

Front kolejki.

Atrybuty chronione

tablica1D< Typ > Tablica

4.5.1 Opis szczegółowy

template < class Typ> class Queue < Typ>

Kolejka jest strukturą typu FIFO (First-In-Fist-Out), zaimplementowana na tablicy dynamicznej Definicja w linii 19 pliku Queue.hh.

4.5.2 Dokumentacja funkcji składowych

```
4.5.2.1 template < class Typ > virtual Typ Queue < Typ >::dequeue ( ) throw EmptyQueueException) [inline], [virtual]
```

Usuwa element z poczatku kolejki

Wyjątki

EmptyQueueException	wyjątek pustej kolejki

Zwraca

Zwraca usunięty element

Implementuje IQueue < Typ >.

Definicja w linii 63 pliku Queue.hh.

4.5.2.2 template < class Typ > virtual void Queue < Typ >::enqueue (Typ item) [inline], [virtual]

Umieszcza element na końcu kolejki

Parametry

in	item	- element, który chcemy umieścić w kolejce
----	------	--

Implementuje IQueue < Typ >.

Definicja w linii 52 pliku Queue.hh.

4.5.2.3 template < class Typ > virtual Typ Queue < Typ >::front () throw EmptyQueueException) [inline], [virtual]

Sprawdza co znajduje się na przodzie kolejki

Wyjątki

EmptyQueueException	wyjątek pustej kolejki

Zwraca

Zwraca pierwszy element w kolejce, bez jego usuwania

Implementuje IQueue < Typ >.

Definicja w linii 78 pliku Queue.hh.

4.5.2.4 template < class Typ > virtual bool Queue < Typ >::isEmpty() [inline], [virtual]

Sprawdza czy w kolejce znajdują się elementy

Zwracane wartości

true	- kolejka pusty
false	- w kolejce są elementy

Implementuje IQueue < Typ >.

Definicja w linii 40 pliku Queue.hh.

4.5.2.5 template < class Typ > virtual int Queue < Typ >::size() [inline], [virtual]

Rozmiar kolejki jest liczbą całkowitą liczoną od 0, gdy stos pusty,

Zwraca

Zwraca liczbę elementów zapisanych w kolejce

Implementuje IQueue < Typ >.

Definicja w linii 31 pliku Queue.hh.

Dokumentacja dla tej klasy została wygenerowana z pliku:

· Queue.hh

4.6 Dokumentacja szablonu klasy Stack< Typ>

Szablon stosu.

#include <Stack.hh>

Diagram dziedziczenia dla Stack< Typ >

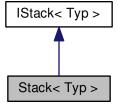
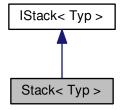


Diagram współpracy dla Stack< Typ >:



Metody publiczne

• virtual int size ()

Rozmiar stosu.

• virtual bool is Empty ()

Czy pusty?

virtual void push (Typ item)

Wkładanie na stos.

• virtual Typ pop () throw (EmptyStackException)

Ściąganie ze stosu.

virtual Typ top () throw (EmptyStackException)
 Szczyt stosu.

Atrybuty chronione

tablica1D
 Typ > Tablica

4.6.1 Opis szczegółowy

```
template < class Typ > class Stack < Typ >
```

Stos jest strukturą typu LIFO (Last-In-Fist-Out), zaimplementowany na tablicy dynamicznej Definicja w linii 16 pliku Stack.hh.

4.6.2 Dokumentacja funkcji składowych

```
4.6.2.1 template < class Typ > virtual bool Stack < Typ >::isEmpty() [inline], [virtual]
```

Sprawdza czy na stosie znajdują się elementy

Zwracane wartości

true	- stos pusty
false	- na stosie są elementy

Implementuje IStack< Typ >.

Definicja w linii 38 pliku Stack.hh.

4.6.2.2 template
$$<$$
 class Typ $>$ virtual Typ Stack $<$ Typ $>$::pop () throw EmptyStackException) [inline], [virtual]

Usuwa element ze szczytu stosu

Wyjątki

EmptyStackException	wyjątek pustego stosu

Zwraca

Zwraca usunięty element

Implementuje IStack< Typ >.

Definicja w linii 62 pliku Stack.hh.

```
4.6.2.3 template < class Typ > virtual void Stack < Typ >::push ( Typ item ) [inline], [virtual]
```

Wkłada element na szczyt stosu

Parametry

in	item	- element, który chcemy umieścić na stosie

Implementuje IStack< Typ >.

Definicja w linii 50 pliku Stack.hh.

4.6.2.4 template < class Typ > virtual int Stack < Typ >::size() [inline], [virtual]

Rozmiar stosu jest liczbą całkowitą liczoną od 0, gdy stos pusty,

Zwraca

Zwraca liczbę elementów zapisanych na stosie

Implementuje IStack< Typ >.

Definicja w linii 28 pliku Stack.hh.

4.6.2.5 template < class Typ > virtual Typ Stack < Typ >::top () throw EmptyStackException) [inline], [virtual]

Element na szcycie stosu

Wyjątki

EmptyStackException	wyjątek pustego stosu

Zwraca

Zwraca element ze szczytu, bez jego usuwania

Implementuje IStack< Typ >.

Definicja w linii 77 pliku Stack.hh.

Dokumentacja dla tej klasy została wygenerowana z pliku:

· Stack.hh

4.7 Dokumentacja klasy Stoper

Diagram dziedziczenia dla Stoper

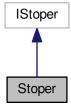
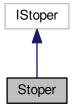


Diagram współpracy dla Stoper:



Metody publiczne

- virtual void start ()
- virtual void stop ()
- virtual double getElapsedTime ()
- virtual bool dumpToFile (ofstream &Plik)

4.7.1 Opis szczegółowy

Definicja w linii 6 pliku Stoper.hh.

Dokumentacja dla tej klasy została wygenerowana z plików:

- · Stoper.hh
- · Stoper.cpp

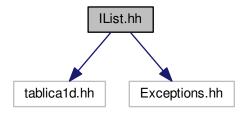
Rozdział 5

Dokumentacja plików

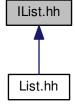
5.1 Dokumentacja pliku IList.hh

Definicja klasy IList.

```
#include "tablicald.hh"
#include "Exceptions.hh"
Wykres zależności załączania dla IList.hh:
```



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



22 Dokumentacja plików

Komponenty

class IList < Typ >
 Interfejs listy.

5.1.1 Opis szczegółowy

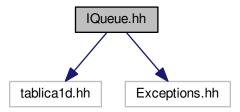
Plik zawiera definicje interfejsu listy

Definicja w pliku IList.hh.

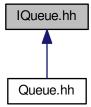
5.2 Dokumentacja pliku IQueue.hh

Definicja klasy IQueue.

```
#include "tablicald.hh"
#include "Exceptions.hh"
Wykres zależności załączania dla IQueue.hh:
```



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

class IQueue < Typ >
 Interfejs kolejki.

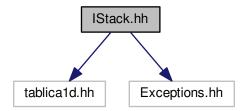
5.2.1 Opis szczegółowy

Plik zawiera klase abstrakcyjną, która jest interfejsem kolejki Definicja w pliku IQueue.hh.

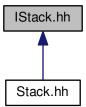
5.3 Dokumentacja pliku IStack.hh

Definicja klasy IStack.

```
#include "tablicald.hh"
#include "Exceptions.hh"
Wykres zależności załączania dla IStack.hh:
```



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

class IStack
 Typ >
 Interfejs stosu.

5.3.1 Opis szczegółowy

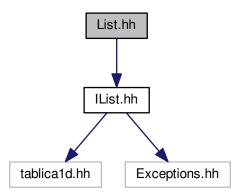
Plik zawiera definicje szablonu klasy IStack, który jest interfejsem stosu Definicja w pliku IStack.hh. 24 Dokumentacja plików

5.4 Dokumentacja pliku List.hh

Definicja klasy List.

#include "IList.hh"

Wykres zależności załączania dla List.hh:



Komponenty

class List< Typ >

Szablon listy.

5.4.1 Opis szczegółowy

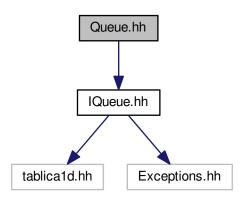
Plik zawiera definicje szablonu klasy IList, który jest interfejsem listy Definicja w pliku List.hh.

5.5 Dokumentacja pliku Queue.hh

Definicja klasy Queue.

#include "IQueue.hh"

Wykres zależności załączania dla Queue.hh:



Komponenty

class Queue < Typ >

Szablon kolejki.

5.5.1 Opis szczegółowy

Plik zawiera definicje szablonu klasy lQueue, który jest interfejsem kolejki Definicja w pliku Queue.hh.

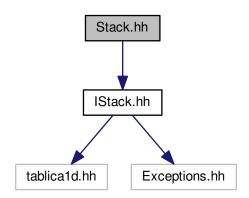
5.6 Dokumentacja pliku Stack.hh

Definicja klasy Stack W pliku znajduje sie klasa Stack.

26 Dokumentacja plików

#include "IStack.hh"

Wykres zależności załączania dla Stack.hh:



Komponenty

class Stack
 Typ >
 Szablon stosu.

Skorowidz

add List, 11
dequeue Queue, 14
enqueue Queue, 15
front Queue, 15
get List, 11
IList< Typ >, 7 IList.hh, 21 IQueue< Typ >, 8 IQueue.hh, 22 IStack< Typ >, 8 IStack.hh, 23 isEmpty List, 11 Queue, 15 Stack, 17
List add, 11 get, 11 isEmpty, 11 remove, 11 size, 13 List< Typ >, 9 List.hh, 24
pop Stack, 17 push Stack, 17
Queue dequeue, 14 enqueue, 15 front, 15 isEmpty, 15 size, 15 Queue < Typ >, 13 Queue.hh, 24
remove List, 11

```
size
List, 13
Queue, 15
Stack, 18

Stack
isEmpty, 17
pop, 17
push, 17
size, 18
top, 18

Stack< Typ >, 16
Stack.hh, 25
Stoper, 18

top
Stack, 18
```