PAMSI LAB II

0.3

Wygenerowano przez Doxygen 1.8.6

Cz, 19 mar 2015 11:38:26

Spis treści

Rozdział 1

Strona główna

Autor

Bartłomiej Ankowski

Data

19.03.2015

Wersja

0.3

1.1 Programu

Rozdział 2

Indeks hierarchiczny

2.1 Hierarchia klas

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

Rozdział 3

Indeks klas

3.1 Lista klas

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

Rozdział 4

Indeks plików

4.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

Rozdział 5

Dokumentacja klas

5.1 Dokumentacja klasy BenchmarkInterfejs

Modeluje pojecie Interfejsu Benchmark'u.

`#include <BenchmarkInterfejs.h>`

Diagram dziedziczenia dla BenchmarkInterfejs

Metody publiczne

- virtual void _Test (const unsigned int Ilosc)=0

  Metoda Wykonujaca pojedyncza operacje.
- virtual void _Wczytaj (string PlikIn, const unsigned n)=0

  Metoda wczytujaca dane z pliku Metoda ma za zadanie wczytac dane z pliku wejsciowego.
- void _WykonajTest (const unsigned int Ilosc_Pow)

  Metoda wykonujaca test odpowiedniej struktury.
- void _Generator (string PlikOut, int n)
- virtual void _Zwolnij (const unsigned int n)=0

  Metoda zwalniajaca pamiec.

5.1.1 Opis szczegółowy

Modeluje pojecie Interfejsu Benchmark'u.

Klasa bazowa dla implementowania benchmarku dla kolejnych struktur danych

Definicja w linii 27 pliku BenchmarkInterfejs.h.

5.1.2 Dokumentacja funkcji składowych

5.1.2.1 void BenchmarkInterfejs::_Generator (string *PlikOut,* int *n*)

Metoda generujaca dane

Metoda ma za zadanie wygenerowac plik z danymi z przedzialu (1-99), jest wywolywana gdy uzytkownik nie poda w argmumencie wywolania programu nazwy pliku wejsciowego z danymi

Parametry

in	*PlikOut*	- Nazwa plik w ktorym zapisywane sa wygenerowane dane
in	*n*	- Ilosc wygenerowanych danych

Definicja w linii 30 pliku BenchmarkInterfejs.cpp.

Oto graf wywoływań tej funkcji:

5.1.2.2 virtual void BenchmarkInterfejs::_Test (const unsigned int *Ilosc*) [pure virtual]

Metoda Wykonujaca pojedyncza operacje.

Metoda ma za zadanie wykonan pojedyncza operacja, ktorej czas jest rejestrowany

Parametry

in	*Ilosc*	- Liczba danych poddana testowi

Implementowany w StrukturyBenchmark.

5.1.2.3 virtual void BenchmarkInterfejs::_Wczytaj (string *PlikIn,* const unsigned *n*) [pure virtual]

Metoda wczytujaca dane z pliku Metoda ma za zadanie wczytac dane z pliku wejsciowego.

Parametry

in	*PlikIn*	- Nazwa pliku wejsciowego
in	*n*	- liczba wczytywanych danych

5.1.2.4 void BenchmarkInterfejs::_WykonajTest (const unsigned int *Ilosc_Pow*)

Metoda wykonujaca test odpowiedniej struktury.

Metoda ma za zadanie wykonac Benchmark dla struktury, dla ustawionej ilosci danych i okreslona przez argument metody ilosc powtorzen.

Parametry

in	*Ilosc_Pow*	- okresla ile razy ma sie wykonac test

Definicja w linii 11 pliku BenchmarkInterfejs.cpp.

Oto graf wywoływań tej funkcji:

5.1.2.5 virtual void BenchmarkInterfejs::_Zwolnij (const unsigned int *n*) [pure virtual]

Metoda zwalniajaca pamiec.

Metoda ma zazadanie wykonac operacje zwalaniania pamieci

Parametry

in	*n*	- liczba danych ktora bedzie zwolniona z pamieci

Implementowany w StrukturyBenchmark.

Dokumentacja dla tej klasy została wygenerowana z plików:

- BenchmarkInterfejs.h
- BenchmarkInterfejs.cpp

5.2 Dokumentacja klasy Lista

`#include <Lista.h>`

Diagram dziedziczenia dla Lista

Diagram współpracy dla Lista:

Metody publiczne

- void Push_Dowolny (int wart, int poz)
  Metoda dodajaca Wezel.
- void Pop_Dowolny (int Pozycja)
  Metoda usuwajaca Wezel.
- void Push (int k)
  Metoda dodajaca wezel.
- void Pop ()
- unsigned int Rozmiar ()
  Metoda informujaca o ilosci wezlow.
- Lista ()
  Konstruktor.
- ~Lista ()
  Destruktor Usuwa wskaznik.
- Lista (const Lista &A)
  Konstruktor Kopiujacy.
- void Pokaz ()
  Metoda wyswietlajaca elementy Listy.

Atrybuty prywatne

- Wezel * Glowa
  Pole klasy Lista Wskaznik na nowo dodany Wezel.
- unsigned int _Ilosc
  Pole klasy Lista Pole przechowuje bierzaca ilosc elementow listy.

5.2.1 Opis szczegółowy

Definicja w linii 13 pliku Lista.h.

5.2.2 Dokumentacja konstruktora i destruktora

5.2.2.1 Lista::Lista ( ) [inline]

Konstruktor.

Konstruktor ustawia wskaznik na NULL i zeruje ilosc wezlow listy

Definicja w linii 70 pliku Lista.h.

5.2.2.2 Lista::~Lista ( ) [inline]

Destruktor Usuwa wskaznik.

Definicja w linii 75 pliku Lista.h.

5.2.2.3 Lista::Lista (const Lista & *A*)

Konstruktor Kopiujacy.

Definicja w linii 78 pliku Lista.cpp.

Oto graf wywołań dla tej funkcji:

5.2.3 Dokumentacja funkcji składowych

5.2.3.1 void Lista::Pokaz ( ) [virtual]

Metoda wyswietlajaca elementy Listy.

Metoda ma za zadanie wyswietlic wszsytkie warotsci znajdujace sie na Liscie

Implementuje Struktury.

Definicja w linii 63 pliku Lista.cpp.

5.2.3.2 int Lista::Pop ( ) [virtual]

Metoda usuwajaca Wezel

Metoda ma za zadanie usunac pierwszy Wezel listy

Implementuje Struktury.

Definicja w linii 96 pliku Lista.cpp.

5.2.3.3 void Lista::Pop_Dowolny (int *Pozycja*)

Metoda usuwajaca Wezel.

Metoda ma za zadanie usunac wezel zgodny z argumentem metody Wezel zosatnie usuniety, a sasiednie elementy zostana polaczaone wskazniekiem

Parametry

in	*Pozycja*	- Numer Wezla ktory zostanie usuniety

Definicja w linii 8 pliku Lista.cpp.

5.2.3.4 void Lista::Push (int *k*) [virtual]

Metoda dodajaca wezel.

Metoda ma za zadanie dodac element na poczatek listy

Parametry

in	*k*	- Wartosc ktora bedzie zapisana w Wezle

Implementuje Struktury.

Definicja w linii 87 pliku Lista.cpp.

Oto graf wywoływań tej funkcji:

5.2.3.5 void Lista::Push_Dowolny (int *wart,* int *poz*)

Metoda dodajaca Wezel.

Metoda ma za zadanie dodac element do listy w zaleznosci od argumentu miejscu i usatwic wartosc zadana przez argument

Parametry

in	*wart*	- Wartosc jaka zostana dodana do Wezla
in	*poz*	- Pozycja w ktorej zosatnie dodany Wezel

Definicja w linii 33 pliku Lista.cpp.

Oto graf wywołań dla tej funkcji:

5.2.3.6 unsigned int Lista::Rozmiar ( ) [inline], [virtual]

Metoda informujaca o ilosci wezlow.

Metoda zwraca informajce o ilosci aktualnych wezlow listy

Zwraca

- Ilosc elementow listy

Implementuje Struktury.

Definicja w linii 64 pliku Lista.h.

Oto graf wywoływań tej funkcji:

5.2.4 Dokumentacja atrybutów składowych

5.2.4.1 unsigned int Lista::_Ilosc [private]

Pole klasy Lista Pole przechowuje bierzaca ilosc elementow listy.

Definicja w linii 25 pliku Lista.h.

5.2.4.2 Wezel* Lista::Glowa [private]

Pole klasy Lista Wskaznik na nowo dodany Wezel.

Definicja w linii 20 pliku Lista.h.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Lista.h
- Lista.cpp

5.3 Dokumentacja klasy Stos

Modeluje pojecie Stosu.

`#include <Stos.h>`

Diagram dziedziczenia dla Stos

Diagram współpracy dla Stos:

Metody publiczne

- void Push (int k)

Metoda dodajaca nowy Wezel.

- void Pop ()

  Metda usuwajaca wezel.

- unsigned int Rozmiar ()

  Metoda informujaca o obecnej ilosci Wezlow.

- Stos ()

  Konstruktor.

- ∼Stos ()

  Destruktor Obiektu Destruktor przy pomocy funkcji pop usuwa wszystkie elemty ze stosu.

- Stos (const Stos &S)

  Konstruktor Kopiujacy.

- void Pokaz ()

  Metoda wyswietlajaca elementy Stosu.

Atrybuty prywatne

- Wezel ∗ Gora

  Pole klasy Stos Pole jest wskaznikiem na ostatnio dodany Wezel.

- unsigned int _Ilosc

  Pole klasy Stos Pole przechowuje informacje o ilosci wezlow.

5.3.1 Opis szczegółowy

Modeluje pojecie Stosu.

Klasa modeluje pojecie Kolejki, dodajac nowy element na jej koniec i sciagajac ostatnio dodany Wezel

Definicja w linii 20 pliku Stos.h.

5.3.2 Dokumentacja konstruktora i destruktora

5.3.2.1 Stos::Stos ( ) [inline]

Konstruktor.

KOnstruktor ma za zadanie ustawic wskaznik na NULL oraz wyzerowac ilosc elmentow podczas tworzenia obiektu tej klasy

Definicja w linii 64 pliku Stos.h.

5.3.2.2 Stos::∼Stos ( )

Destruktor Obiektu Destruktor przy pomocy funkcji pop usuwa wszystkie elemty ze stosu.

Definicja w linii 20 pliku Stos.cpp.

5.3.2.3 Stos::Stos (const Stos & *S*)

Konstruktor Kopiujacy.

Definicja w linii 9 pliku Stos.cpp.

5.3.3 Dokumentacja funkcji składowych

5.3.3.1 void Stos::Pokaz ( ) [virtual]

Metoda wyswietlajaca elementy Stosu.

Metoda ma za zadanie wyswietlic wszsytkie warotsci znajdujace sie na Stosie

Implementuje Struktury.

Definicja w linii 50 pliku Stos.cpp.

5.3.3.2 int Stos::Pop ( ) [virtual]

Metda usuwajaca wezel.

Metoda ma za zadanie zdjac ostatnio dodany element ze stosu danych

Implementuje Struktury.

Definicja w linii 36 pliku Stos.cpp.

5.3.3.3 void Stos::Push (int *k*) [virtual]

Metoda dodajaca nowy Wezel.

Metoda ma za zadanie dodac nowy wezel na koniec , umiescic w nim warosc zadana jako argument oraz pokazac wskaznkiem na ostatni Wezel przed dodaniem nowego Wezla

Parametry

in	*k*	- Wartosc ktora zostanie umieszczona w odpowiednim polu Wezla

Implementuje Struktury.

Definicja w linii 28 pliku Stos.cpp.

5.3.3.4 unsigned int Stos::Rozmiar ( ) [inline], [virtual]

Metoda informujaca o obecnej ilosci Wezlow.

Metoda zwraca informacje o ilosci obecnie znajdujacych sie elementow na stosie

Zwraca

- Zwraca ilosc elementow na Stosie

Implementuje Struktury.

Definicja w linii 57 pliku Stos.h.

5.3.4 Dokumentacja atrybutów składowych

5.3.4.1 unsigned int Stos::_Ilosc [private]

Pole klasy Stos Pole przechowuje informacje o ilosci wezlow.

Definicja w linii 32 pliku Stos.h.

5.3.4.2 Wezel* Stos::Gora [private]

Pole klasy Stos Pole jest wskaznikiem na ostatnio dodany Wezel.

Definicja w linii 27 pliku Stos.h.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Stos.h
- Stos.cpp

5.4 Dokumentacja klasy Struktury

Modeluje pojecie Struktury danych, klasa bazowa dla Stosu,Kolejki i Listy.

`#include <Struktury.h>`

Diagram dziedziczenia dla Struktury

Komponenty

- struct Wezel

  Modeluje pojecie wezla Struktura przeznaczona do dziedziczenia dla klas pochodnych, opartych o dzialanie listy.

Metody publiczne

- virtual void Push (int k)=0

  Metoda dodajaca kolejny Wezel.
- virtual void Pop ()=0

  Metoda usuwajaca Wezel.
- virtual unsigned int Rozmiar ()=0

  Metoda zwracajaca rozmiar Struktury.
- virtual void Pokaz ()=0

  Metoda wyswietlajaca dane.

5.4.1 Opis szczegółowy

Modeluje pojecie Struktury danych, klasa bazowa dla Stosu,Kolejki i Listy.

Definicja w linii 19 pliku Struktury.h.

5.4.2 Dokumentacja funkcji składowych

5.4.2.1 virtual void Struktury::Pokaz ( ) [pure virtual]

Metoda wyswietlajaca dane.

Metoda ma za zadanie wyswietlic wszytskie dane nalezace do struktury

Implementowany w Lista i Stos.

5.4.2.2 virtual void Struktury::Pop ( ) [pure virtual]

Metoda usuwajaca Wezel.

Metoda ma za zadanie usunac wezel i w zaleznosci od implementowanej struktury bedzie to usuwany element z poczatku lub konca listy Wezlow

Implementowany w Lista i Stos.

5.4.2.3 virtual void Struktury::Push (int *k*) [pure virtual]

Metoda dodajaca kolejny Wezel.

Metoda ma za zadanie dodac kolejny wezel do naszej struktury oraz zapisac w nim odpowiednia wartosc.W zaleznosci od implementowanej struktury element bedzie dodawany na poczatku lub na koncu listy.

Parametry

in	*k*	- wartosc typu calkowitnego, ktora bedzie umieszona w wezle

Implementowany w Lista i Stos.

5.4.2.4 virtual unsigned int Struktury::Rozmiar ( ) [pure virtual]

Metoda zwracajaca rozmiar Struktury.

Metoda ma zadanie zwrocic bierzaca liczbe elementow nalezacych do danej struktury

Zwraca

- Bierzaca liczba elementow Struktury danych

Implementowany w Lista i Stos.

Dokumentacja dla tej klasy została wygenerowana z pliku:

- Struktury.h

5.5 Dokumentacja klasy StrukturyBenchmark

`#include <StrukturyBenchmark.h>`

Diagram dziedziczenia dla StrukturyBenchmark

Diagram współpracy dla StrukturyBenchmark:

Metody publiczne

- void _Test (const unsigned int n)

  Metoda wykonujaca test dla odpowiedniej struktury.
- void _Wczytaj (string PlikIn, const unsigned int Ilosc)

  Metoda Wczytujaca dane.
- void _Ustaw (Struktury &K)

  Metoda przypisujaca obiekt danej struktury.
- StrukturyBenchmark ()

  Konstruktor.
- void _Przydziel (const unsigned int n)

  Metoda alokujaca pamiec.
- virtual ~StrukturyBenchmark ()

  Destruktor.
- void _Zwolnij (const unsigned int n)

  Metoda zwalniajaca zaalokowana przez struktury pamiec.

Atrybuty prywatne

- Struktury * S

  Pole StrulturyBenchmark Pole zawiera wskaźnik na Struktury, za pomoca niego i metod wirtualnych beda wywolywane odpowiednie dla danej strktury metody.

- int * W

  Pole StrkturyBenchmark Pole zawiera wskaznik na typ calkowity, sluzy on do alokowania pamieci dla wczytanych z pliku danych.

5.5.1 Opis szczegółowy

Klasa modeluje pojecie Benchmarku przeznaczonego dla struktur danych przechowujace dane

Definicja w linii 17 pliku StrukturyBenchmark.h.

5.5.2 Dokumentacja konstruktora i destruktora

5.5.2.1 StrukturyBenchmark::StrukturyBenchmark ( ) [inline]

Konstruktor.

Konstruktor sprawia ze wskazniki nowo powstalego obiektu wskazuja na NULL

Definicja w linii 63 pliku StrukturyBenchmark.h.

5.5.2.2 virtual StrukturyBenchmark::~StrukturyBenchmark ( ) [inline], [virtual]

Destruktor.

Definicja w linii 76 pliku StrukturyBenchmark.h.

5.5.3 Dokumentacja funkcji składowych

5.5.3.1 void StrukturyBenchmark::_Przydziel (const unsigned int *n*)

Metoda alokujaca pamiec.

Metoda ma za zadanie zaalokowac odpowiednia ilosc danych w zaleznosci od tego ile zostalo ich wczytanych

Parametry

in	*n*	- Ilosc wczytanych danych

Definicja w linii 14 pliku StrukturyBenchmark.cpp.

Oto graf wywoływań tej funkcji:

5.5.3.2 void StrukturyBenchmark::_Test (const unsigned int *n*) [virtual]

Metoda wykonujaca test dla odpowiedniej struktury.

Metoda ma za zadanie wykonac zapelnienie struktury danymi o zadanej w argumencie ilosci

Parametry

in	*n*	- ilosc danych ktora zapelnona struktura

Implementuje BenchmarkInterfejs.

Definicja w linii 21 pliku StrukturyBenchmark.cpp.

5.5.3.3 void StrukturyBenchmark::_Ustaw (Struktury & *K*) [inline]

Metoda przypisujaca obiekt danej struktury.

Metoda ma za zadanie pokazywac na testowana obecnie strukture danych, aby za pomoca wskaznika wywolywac metody z odpowiednich struktur

Parametry

in	*K*	- Adres aktualnie testowanego obiektu danej struktury

Definicja w linii 57 pliku StrukturyBenchmark.h.

Oto graf wywoływań tej funkcji:

5.5.3.4 void StrukturyBenchmark::_Wczytaj (string *PlikIn,* const unsigned int *Ilosc*)

Metoda Wczytujaca dane.

Metoda ma za zadanie wczytac dane wejciowe o podanej przez argument nazwie oraz przypisac wskaznik do zaalokwanych w pamieci danych

Parametry

in	*PlikIn*	- nazwa pliku wejsciowego z danymi
in	*Ilosc*	- Ilosc danych jaka bedzie wczytywana

Definicja w linii 27 pliku StrukturyBenchmark.cpp.

Oto graf wywołań dla tej funkcji:

Oto graf wywoływań tej funkcji:

5.5.3.5 void StrukturyBenchmark::_Zwolnij (const unsigned int *n*) [virtual]

Metoda zwalniajaca zaalokowana przez struktury pamiec.

Metoda ma za zadanie oproznic zaladowane do struktury dane,

Implementuje BenchmarkInterfejs.

5.5.4 Dokumentacja atrybutów składowych

5.5.4.1 Struktury* StrukturyBenchmark::S [private]

Pole StrulturyBenchmark Pole zawiera wskaźnik na Struktury, za pomoca niego i metod wirtualnych beda wywolywane odpowiednie dla danej strktury metody.

Definicja w linii 25 pliku StrukturyBenchmark.h.

5.5.4.2 int* StrukturyBenchmark::W [private]

Pole StrkturyBenchmark Pole zawiera wskaznik na typ calkowity, sluzy on do alokowania pamieci dla wczytanych z pliku danych.

Definicja w linii 31 pliku StrukturyBenchmark.h.

Dokumentacja dla tej klasy została wygenerowana z plików:

- StrukturyBenchmark.h
- StrukturyBenchmark.cpp

5.6 Dokumentacja struktury Struktury::Wezel

Modeluje pojecie wezla Struktura przeznaczona do dziedziczenia dla klas pochodnych, opartych o dzialanie listy.

`#include <Struktury.h>`

Diagram współpracy dla Struktury::Wezel:

Atrybuty publiczne

- Wezel * _Nast

  Pole Wezla Pole bedace wskaznikiem na kolejny element listy.
- int _Wartosc

  Pole Wezla Pole przechowuje wartoc typu calkowitego.

5.6.1 Opis szczegółowy

Modeluje pojecie wezla Struktura przeznaczona do dziedziczenia dla klas pochodnych, opartych o dzialanie listy.

Definicja w linii 26 pliku Struktury.h.

5.6.2 Dokumentacja atrybutów składowych

5.6.2.1 Wezel* Struktury::Wezel::_Nast

Pole Wezla Pole bedace wskaznikiem na kolejny element listy.

Definicja w linii 31 pliku Struktury.h.

5.6.2.2 int Struktury::Wezel::_Wartosc

Pole Wezla Pole przechowuje wartoc typu calkowitego.

Definicja w linii 35 pliku Struktury.h.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- Struktury.h

Rozdział 6

Dokumentacja plików

6.1 Dokumentacja pliku BenchmarkInterfejs.cpp

Definicje Metod klasy BenchmarkInterfejs.

```
#include "BenchmarkInterfejs.h"
#include <sys/time.h>
#include <iostream>
#include <fstream>
```

Wykres zależności załączania dla BenchmarkInterfejs.cpp:

6.1.1 Opis szczegółowy

Definicje Metod klasy BenchmarkInterfejs.

Definicja w pliku BenchmarkInterfejs.cpp.

6.2 Dokumentacja pliku BenchmarkInterfejs.h

```
#include <iostream>
#include <cstdlib>
#include <fstream>
#include <cstring>
#include "Struktury.h"
```

Wykres zależności załączania dla BenchmarkInterfejs.h: Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:

Komponenty

- class BenchmarkInterfejs

  Modeluje pojecie Interfejsu Benchmark'u.

Definicje

- #define ILOSC 4

6.2.1 Dokumentacja definicji

6.2.1.1 #define ILOSC 4

Definicja w linii 3 pliku BenchmarkInterfejs.h.

6.3 Dokumentacja pliku Kolejka.cpp

Definicje Metod klasy Kolejka.

```
#include "Kolejka.h"
```

Wykres zależności załączania dla Kolejka.cpp:

6.3.1 Opis szczegółowy

Definicje Metod klasy Kolejka.

Definicja w pliku Kolejka.cpp.

6.4 Dokumentacja pliku Kolejka.h

```
#include <iostream>
#include <cstdlib>
#include "Struktury.h"
```

Wykres zależności załączania dla Kolejka.h: Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:

6.5 Dokumentacja pliku Lista.cpp

Definicje Metod klasy Lista.

```
#include "Lista.h"
```

Wykres zależności załączania dla Lista.cpp:

6.5.1 Opis szczegółowy

Definicje Metod klasy Lista.

Definicja w pliku Lista.cpp.

6.6 Dokumentacja pliku Lista.h

```
#include <iostream>
#include <cstdlib>
#include "Struktury.h"
```

Wykres zależności załączania dla Lista.h: Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:

Komponenty

- class Lista

6.7 Dokumentacja pliku Main.cpp

funkcja glowna programu

```
#include "Lista.h"
#include "Stos.h"
#include "Kolejka.h"
#include "StrukturyBenchmark.h"
```

Wykres zależności załączania dla Main.cpp:

Definicje

- #define ILOSC_DANYCH 100000000

Funkcje

- int main (int argc, char ∗argv[])

6.7.1 Opis szczegółowy

funkcja glowna programu

Definicja w pliku Main.cpp.

6.7.2 Dokumentacja definicji

6.7.2.1 #define ILOSC_DANYCH 100000000

Definicja w linii 5 pliku Main.cpp.

6.7.3 Dokumentacja funkcji

6.7.3.1 int main (int *argc,* char ∗ *argv[]*)

Definicja w linii 10 pliku Main.cpp.

Oto graf wywołań dla tej funkcji:

6.8 Dokumentacja pliku Stos.cpp

Definicje metod klasy Stos.

```
#include "Stos.h"
```

Wykres zależności załączania dla Stos.cpp:

6.8.1 Opis szczegółowy

Definicje metod klasy Stos.

Definicja w pliku Stos.cpp.

6.9 Dokumentacja pliku Stos.h

```
#include <iostream>
#include "Struktury.h"
```
Wykres zależności załączania dla Stos.h: Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:

Komponenty

- class Stos

  Modeluje pojecie Stosu.

6.10 Dokumentacja pliku strona-glowna.dox

6.11 Dokumentacja pliku Struktury.h

```
#include <iostream>
#include <cstring>
#include <sstream>
#include <fstream>
```
Wykres zależności załączania dla Struktury.h: Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:

Komponenty

- class Struktury

  Modeluje pojecie Struktury danych, klasa bazowa dla Stosu,Kolejki i Listy.
- struct Struktury::Wezel

  Modeluje pojecie wezla Struktura przeznaczona do dziedziczenia dla klas pochodnych, opartych o dzialanie listy.

6.12 Dokumentacja pliku StrukturyBenchmark.cpp

Definicje metod klasy StrukturyBenchmark.

```
#include "StrukturyBenchmark.h"
```
Wykres zależności załączania dla StrukturyBenchmark.cpp:

6.12.1 Opis szczegółowy

Definicje metod klasy StrukturyBenchmark.

Definicja w pliku StrukturyBenchmark.cpp.

6.13 Dokumentacja pliku StrukturyBenchmark.h

```
#include "BenchmarkInterfejs.h"
#include "Struktury.h"
```
Wykres zależności załączania dla StrukturyBenchmark.h: Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:

Komponenty

- class StrukturyBenchmark

Skorowidz