PAMSI LAB II

0.3

Wygenerowano przez Doxygen 1.8.6

Śr, 25 mar 2015 22:00:02

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 32 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 Kolejka

Modeluje pojecie Kolejki.

`#include <Kolejka.h>`

Diagram dziedziczenia dla Kolejka

Diagram współpracy dla Kolejka:

Komponenty

- struct Wezel

Metody publiczne

- Kolejka ()

  Konstruktor.
- ~Kolejka ()

  Destruktor.
- void _Pokaz ()

  Metoda wyswietlajaca elementu Kolejki.
- void _Push (int k, unsigned int Pozycja=0)

  Metoda dodajaca nowy Wezel.
- int _Pop (unsigned int Pozycja=0)

  Metda usuwajaca wezel.
- unsigned int _Rozmiar ()

  Metoda informujaca o obecnej ilosci Wezlow.
- void _Zwolnij ()

  Metoda zwalniajaca pamiec zajeta przez struktre.

Atrybuty prywatne

- Wezel * _Pierwszy

  Pole klasy Kolejka.
- Wezel * _Ostatni

  Pole klasy Kolejka.
- unsigned int _Ilosc

  Pole klasy Kolejka.

5.2.1 Opis szczegółowy

Modeluje pojecie Kolejki.

Klasa modeluje pojecie kolejki, dodajac nowy element na jej koniec i sciagajac pierwszy doddany element

Definicja w linii 22 pliku Kolejka.h.

5.2.2 Dokumentacja konstruktora i destruktora

5.2.2.1 Kolejka::Kolejka ( ) [inline]

Konstruktor.

Ustawia wskazniki na NULL oraz zeruje ilosc elementow przy tworzeniu obiektow danej klasy

Definicja w linii 66 pliku Kolejka.h.

5.2.2.2 Kolejka::~Kolejka ( )

Destruktor.

Definicja w linii 66 pliku Kolejka.cpp.

Oto graf wywołań dla tej funkcji:

5.2.3 Dokumentacja funkcji składowych

5.2.3.1 void Kolejka::_Pokaz ( ) [virtual]

Metoda wyswietlajaca elementu Kolejki.

Metoda ma za zadanie wyswietlic wszsytkie warotsci znajdujace sie w kolejce

Implementuje Struktury.

Definicja w linii 51 pliku Kolejka.cpp.

5.2.3.2 int Kolejka::_Pop (unsigned int *Pozycja* = 0) [virtual]

Metda usuwajaca wezel.

Metoda ma za zadanie usunac pierwszy dodany element z kolejki oraz ustawic wskaznik przed ostatniego elementu na NULL

Implementuje Struktury.

Definicja w linii 38 pliku Kolejka.cpp.

Oto graf wywoływań tej funkcji:

5.2.3.3 void Kolejka::_Push (int *k*, unsigned int *Pozycja* = 0) [virtual]

Metoda dodajaca nowy Wezel.

Metoda ma za zadanie dodac nowy wezel na koniec kolejki, 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 19 pliku Kolejka.cpp.

5.2.3.4 unsigned int Kolejka::_Rozmiar ( ) [inline], [virtual]

Metoda informujaca o obecnej ilosci Wezlow.

Metoda zwraca informacje o ilosci obecnie znajdujacych sie elementow w kolejce

Zwraca

- Zwraca ilosc elementow w kolejce

Implementuje Struktury.

Definicja w linii 102 pliku Kolejka.h.

5.2.3.5 void Kolejka::_Zwolnij ( ) [virtual]

Metoda zwalniajaca pamiec zajeta przez struktre.

Metoda ma za zadanie zwolnij pamiec zajeta przez zaladowane do struktury dane, elementy sa usuwany dopoki wskaznik pokazujacy na poczatek kolejki nie bedzie wskazywal na NULL

Implementuje Struktury.

Definicja w linii 6 pliku Kolejka.cpp.

5.2.4 Dokumentacja atrybutów składowych

5.2.4.1 unsigned int Kolejka::_Ilosc [private]

Pole klasy Kolejka.

Pole przechowuje inrofmracje o ilosci obecnie znajdujacych sie elementow w kolejce

Definicja w linii 58 pliku Kolejka.h.

5.2.4.2 Wezel* Kolejka::_Ostatni [private]

Pole klasy Kolejka.

Pole jest wskaznikiem na ostatni element kolejki

Definicja w linii 51 pliku Kolejka.h.

5.2.4.3 Wezel* Kolejka::_Pierwszy [private]

Pole klasy Kolejka.

Pole jest wskaznikiem na pierwszy element kolejki

Definicja w linii 45 pliku Kolejka.h.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Kolejka.h
- Kolejka.cpp

5.3 Dokumentacja klasy Lista

`#include <Lista.h>`

Diagram dziedziczenia dla Lista

Diagram współpracy dla Lista:

Komponenty

- struct Wezel

Metody publiczne

- void _Push (int wart, unsigned int poz=1)

  Metoda dodajaca Wezel.
- int _Pop (unsigned int Pozycja=1)

  Metoda usuwajaca Wezel.
- unsigned int _Rozmiar ()

  Metoda informujaca o ilosci wezlow.
- Lista ()

  Konstruktor.
- ~Lista ()

  Destruktor Usuwa wskaznik.
- void _Pokaz ()

  Konstruktor Kopiujacy.
- void _Zwolnij ()

  Metoda zwalniajaca pamiec zajeta przez struktre.

Atrybuty prywatne

- Wezel * Glowa

  Pole klasy Lista Wskaznik na nowo dodany Wezel.
- unsigned int _Ilosc

  Pole klasy Lista Pole przechowuje bierzaca ilosc elementow listy.

5.3.1 Opis szczegółowy

Definicja w linii 12 pliku Lista.h.

5.3.2 Dokumentacja konstruktora i destruktora

5.3.2.1 Lista::Lista ( ) [inline]

Konstruktor.

Konstruktor ustawia wskaznik na NULL i zeruje ilosc wezlow listy

Definicja w linii 72 pliku Lista.h.

5.3.2.2 Lista::~Lista ( ) [inline]

Destruktor Usuwa wskaznik.

Definicja w linii 77 pliku Lista.h.

5.3.3 Dokumentacja funkcji składowych

5.3.3.1 void Lista::_Pokaz ( ) [virtual]

Konstruktor Kopiujacy.

Metoda wyswietlajaca elementy Listy

Metoda ma za zadanie wyswietlic wszsytkie warotsci znajdujace sie na Liscie

Implementuje Struktury.

Definicja w linii 90 pliku Lista.cpp.

5.3.3.2 int Lista::_Pop (unsigned int *Pozycja* = 1) [virtual]

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

Implementuje Struktury.

Definicja w linii 22 pliku Lista.cpp.

5.3.3.3 void Lista::_Push (int *wart*, unsigned int *poz* = 1) [virtual]

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

Implementuje Struktury.

Definicja w linii 59 pliku Lista.cpp.

5.3.3.4 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 66 pliku Lista.h.

5.3.3.5 void Lista::_Zwolnij ( ) [virtual]

Metoda zwalniajaca pamiec zajeta przez struktre.

Metoda ma za zadanie zwolnij pamiec zajeta przez zaladowane do struktury dane, elementy sa usuwany dopoki wskaznik pokazujacy na poczatek listy nie bedzie wskazywal na NULL

Implementuje Struktury.

Definicja w linii 9 pliku Lista.cpp.

5.3.4 Dokumentacja atrybutów składowych

5.3.4.1 unsigned int Lista::_Ilosc [private]

Pole klasy Lista Pole przechowuje bierzaca ilosc elementow listy.

Definicja w linii 39 pliku Lista.h.

5.3.4.2 Wezel* Lista::Glowa [private]

Pole klasy Lista Wskaznik na nowo dodany Wezel.

Definicja w linii 34 pliku Lista.h.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Lista.h
- Lista.cpp

5.4 Dokumentacja klasy ListaTab

`#include <ListaTab.h>`

Diagram dziedziczenia dla ListaTab

Diagram współpracy dla ListaTab:

Metody publiczne

- ListaTab ()

  Konstruktor Podczas tworzenia obiektu tej klasy automatycznie alokowana jest tablica o rozmiarze 1 oraz ustawienie obecnej liczby elementow listy na 0.
- ~ListaTab ()

  Konstruktor Kopiujacy.
- void _Pokaz ()

  Metoda wypisujaca elemeny listy.
- int _Pop (unsigned int Pozycja)

  Metoda sciagajaca element z listy.
- void _Push (int k, unsigned int Pozycja)

  Metoda dodajaca elemet do tablicy.
- unsigned int _Rozmiar ()

  Metoda zwracajaca rozmiar listy.
- void _Zwolnij ()

Atrybuty prywatne

- int * _L

  Pole klasy ListaTab.
- unsigned int _RozmiarL

  Pole Klasy ListaTab.
- unsigned int _RozmiarT

  Pole Klasy ListaTab.

5.4.1 Opis szczegółowy

Definicja w linii 13 pliku ListaTab.h.

5.4.2 Dokumentacja konstruktora i destruktora

5.4.2.1 ListaTab::ListaTab ( )

Konstruktor Podczas tworzenia obiektu tej klasy automatycznie alokowana jest tablica o rozmiarze 1 oraz ustawienie obecnej liczby elementow listy na 0.

Definicja w linii 65 pliku ListaTab.cpp.

5.4.2.2 ListaTab::~ListaTab ( ) [inline]

Konstruktor Kopiujacy.

Destruktor obiektu

Definicja w linii 48 pliku ListaTab.h.

5.4.3 Dokumentacja funkcji składowych

5.4.3.1 void ListaTab::_Pokaz ( ) [virtual]

Metoda wypisujaca elemeny listy.

Metoda ma za zadanie wypisac wszystkie elementy znajdujace sie obecnie na liscie danych

Implementuje Struktury.

Definicja w linii 76 pliku ListaTab.cpp.

5.4.3.2 int ListaTab::_Pop (unsigned int *Pozycja*) [virtual]

Metoda sciagajaca element z listy.

Metoda ma za zadanie sciagnac wybrany przez uzytkownika element listy oraz kazdorazowa zmniejszyc tablice z danymi o 1 oraz przekopiowac pozostale elementy listy

Parametry

in	*Pozycja*	- numer elementy kotry zostanie usuniety z listy i zostanie zwrocona jego wartosc

Zwraca

Zwraca wybrany przez uzytkownika element

Implementuje Struktury.

Definicja w linii 13 pliku ListaTab.cpp.

5.4.3.3 void ListaTab::_Push (int *k,* unsigned int *Pozycja*) [virtual]

Metoda dodajaca elemet do tablicy.

Metoda ma za zadanie dodac nowy element w wybranym przez uzytkownika miejscu oraz zwiekszyc kazdorazowa tablice danych o 1 i przepisac pozostale elementy do nowej

Parametry

in	*k*	- wartosc jaka chcemy dodac do listy
in	*Pozycja*	- Pozycja na ktorej chcemy dodac wartosc

Implementuje Struktury.

Definicja w linii 41 pliku ListaTab.cpp.

5.4.3.4 unsigned int ListaTab::_Rozmiar ( ) [inline], [virtual]

Metoda zwracajaca rozmiar listy.

Metoda zwraca informacje o obecnej ilosci danych w strukturze

Zwraca

Zwraca ilosc elementow listy

Implementuje Struktury.

Definicja w linii 87 pliku ListaTab.h.

5.4.3.5 void ListaTab::_Zwolnij ( ) [virtual]

Metoda zwalniajaca pamiec

Metoda ma za zadanie zwolnij pamiec zajmowana przez dane, dopoki ilosc elementow listy nie wynosi 0 wykonywana jest metoda _Pop, aby oproznic stos i zwolnic pamiec

Implementuje Struktury.

Definicja w linii 7 pliku ListaTab.cpp.

5.4.4 Dokumentacja atrybutów składowych

5.4.4.1 int* ListaTab::_L [private]

Pole klasy ListaTab.

Pole zawiera wskaznik na typ calkowity, sluzy do alokacji pamieci na dynamiczna tablice

Definicja w linii 21 pliku ListaTab.h.

5.4.4.2 unsigned int ListaTab::_RozmiarL [private]

Pole Klasy ListaTab.

Pole przechowuje informacje o ilosci obecnie znajdujacych sie elementow na liscie danych

Definicja w linii 27 pliku ListaTab.h.

5.4.4.3 unsigned int ListaTab::_RozmiarT [private]

Pole Klasy ListaTab.

Pole przechowuje informacje o obecnycm rozmiarze tablicy danych

Definicja w linii 33 pliku ListaTab.h.

Dokumentacja dla tej klasy została wygenerowana z plików:

- ListaTab.h
- ListaTab.cpp

5.5 Dokumentacja klasy Stos

Modeluje pojecie Stosu.

`#include <Stos.h>`

Diagram dziedziczenia dla Stos

Diagram współpracy dla Stos:

Komponenty

- struct Wezel

Metody publiczne

- void _Push (int k, unsigned int Pozycja=0)

  Metoda dodajaca nowy Wezel.
- int _Pop (unsigned int Pozycja=0)

  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.
- void _Zwolnij ()

  Metoda zwalniajaca pamiec zajeta przez struktre.

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.

5.5.1 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.

5.5.2 Dokumentacja konstruktora i destruktora

5.5.2.1 Stos::Stos ( ) [inline]

Konstruktor.

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

Definicja w linii 82 pliku Stos.h.

5.5.2.2 Stos::~Stos ( )

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

Definicja w linii 23 pliku Stos.cpp.

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

Konstruktor Kopiujacy.

5.5.3 Dokumentacja funkcji składowych

5.5.3.1 void Stos::_Pokaz ( ) [virtual]

Metoda wyswietlajaca elementy Stosu.

Metoda ma za zadanie wyswietlic wszsytkie warotsci znajdujace sie na Stosie

Implementuje Struktury.

Definicja w linii 55 pliku Stos.cpp.

5.5.3.2 int Stos::_Pop (unsigned int *Pozycja* = 0) [virtual]

Metda usuwajaca wezel.

Metoda ma za zadanie zdjac ostatnio dodany element ze stosu danych oraz zwrocic przechowywana wartosc

Zwraca

Zwraca ostatnia dodana wartosc

Implementuje Struktury.

Definicja w linii 40 pliku Stos.cpp.

5.5.3.3 void Stos::_Push (int *k*, unsigned int *Pozycja* = 0) [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 31 pliku Stos.cpp.

5.5.3.4 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 75 pliku Stos.h.

5.5.3.5 void Stos::_Zwolnij ( ) [virtual]

Metoda zwalniajaca pamiec zajeta przez struktre.

Metoda ma za zadanie zwolnij pamiec zajeta przez zaladowane do struktury dane, elementy sa usuwany dopoki wskaznik pokazujacy na poczatek kolejki nie bedzie wskazywal na NULL

Implementuje Struktury.

Definicja w linii 7 pliku Stos.cpp.

5.5.4 Dokumentacja atrybutów składowych

5.5.4.1 unsigned int Stos::_Ilosc [private]

Pole klasy Stos Pole przechowuje informacje o ilosci wezlow.

Definicja w linii 47 pliku Stos.h.

5.5.4.2 Wezel* Stos::Gora [private]

Pole klasy Stos Pole jest wskaznikiem na ostatnio dodany Wezel.

Definicja w linii 42 pliku Stos.h.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Stos.h
- Stos.cpp

5.6 Dokumentacja klasy StosTab

`#include <StosTab.h>`

Diagram dziedziczenia dla StosTab

Diagram współpracy dla StosTab:

Metody publiczne

- StosTab ()

  Konstruktor Podczas tworzenia obiektu tej klasy automatycznie alokowana jest tablica o rozmiarze 1 oraz ustawienie obecnej liczby elementow listy na 0.
- ~StosTab ()

  Konstruktor Kopiujacy.
- void _Pokaz ()

  Metoda wypisujaca elemeny Stosu.
- int _Pop (unsigned int Pozycja=0)

  Metoda sciagajaca element ze stosu.
- void _Push (int k, unsigned int Pozycja=0)

  Metoda dodajaca elemet do tablicy.
- unsigned int _Rozmiar ()

  Metoda zwracajaca rozmiar listy.
- void _Zwolnij ()

Atrybuty prywatne

- int * _L

  Pole klasy StosTab.
- unsigned int _RozmiarL

  Pole Klasy StosTab.
- unsigned int _RozmiarT

  Pole Klasy StosTab.

5.6.1 Opis szczegółowy

Definicja w linii 13 pliku StosTab.h.

5.6.2 Dokumentacja konstruktora i destruktora

5.6.2.1 StosTab::StosTab ( )

Konstruktor Podczas tworzenia obiektu tej klasy automatycznie alokowana jest tablica o rozmiarze 1 oraz ustawienie obecnej liczby elementow listy na 0.

Definicja w linii 59 pliku StosTab.cpp.

5.6.2.2 StosTab::~StosTab ( ) [inline]

Konstruktor Kopiujacy.

Destruktor obiektu

Definicja w linii 48 pliku StosTab.h.

5.6.3 Dokumentacja funkcji składowych

5.6.3.1 void StosTab::_Pokaz ( ) [virtual]

Metoda wypisujaca elemeny Stosu.

Metoda ma za zadanie wypisac wszystkie elementy znajdujace sie obecnie na liscie danych

Implementuje Struktury.

Definicja w linii 68 pliku StosTab.cpp.

5.6.3.2 int StosTab::_Pop (unsigned int *Pozycja* = 0) [virtual]

Metoda sciagajaca element ze stosu.

Metoda ma za zadanie sciagnac ostatni element stosu, w przypadku gdy tablica jest do połowy pusta nastepuje utworzenie nowej tablicy o dwa razy mniejszym rozmiarze

Parametry

in	*Pozycja*	- numer elementy kotry zostanie usuniety z listy i zostanie zwrocona jego wartosc

Zwraca

Zwraca wybrany przez uzytkownika element

Implementuje Struktury.

Definicja w linii 13 pliku StosTab.cpp.

5.6.3.3 void StosTab::_Push (int *k*, unsigned int *Pozycja* = 0) [virtual]

Metoda dodajaca elemet do tablicy.

Metoda ma za zadanie dodac nowy element na koncu stosu, w przypadku zapelnienia tablicy nastepuje utworzenie nowej tablicy i przepisanie elementow

Parametry

in	*k*	- wartosc jaka chcemy dodac do listy
in	*Pozycja*	- Pozycja na ktorej chcemy dodac wartosc

Implementuje Struktury.

Definicja w linii 39 pliku StosTab.cpp.

5.6.3.4 unsigned int StosTab::_Rozmiar ( ) [inline], [virtual]

Metoda zwracajaca rozmiar listy.

Metoda zwraca informacje o obecnej ilosci danych w strukturze

Zwraca

Zwraca ilosc elementow listy

Implementuje Struktury.

Definicja w linii 86 pliku StosTab.h.

5.6.3.5 void StosTab::_Zwolnij ( ) [virtual]

Metoda zwalniajaca pamiec

Metoda ma za zadanie zwolnij pamiec zajmowana przez dane, dopoki ilosc elementow listy nie wynosi 0 wykonywana jest metoda _Pop, aby oproznic stos i zwolnic pamiec

Implementuje Struktury.

Definicja w linii 7 pliku StosTab.cpp.

5.6.4 Dokumentacja atrybutów składowych

5.6.4.1 int* StosTab::_L [private]

Pole klasy StosTab.

Pole zawiera wskaznik na typ calkowity, sluzy do alokacji pamieci na dynamiczna tablice

Definicja w linii 21 pliku StosTab.h.

5.6.4.2 unsigned int StosTab::_RozmiarL [private]

Pole Klasy StosTab.

Pole przechowuje informacje o ilosci obecnie znajdujacych sie elementow na liscie danych

Definicja w linii 27 pliku StosTab.h.

5.6.4.3 unsigned int StosTab::_RozmiarT [private]

Pole Klasy StosTab.

Pole przechowuje informacje o obecnycm rozmiarze tablicy danych

Definicja w linii 33 pliku StosTab.h.

Dokumentacja dla tej klasy została wygenerowana z plików:

- StosTab.h
- StosTab.cpp

5.7 Dokumentacja klasy Struktury

Modeluje pojecie Struktury danych, klasa bazowa dla Stosu,Kolejki i Listy,zarowno w implemenetacji wskaznikowej jak i tablicowej.

`#include <Struktury.h>`

Diagram dziedziczenia dla Struktury

Metody publiczne

- virtual void _Push (int k, unsigned int Pozycja=1)=0

  Metoda dodajaca kolejny element struktury.
- virtual int _Pop (unsigned int Pozycja=1)=0

  Metoda usuwajaca element.
- virtual unsigned int _Rozmiar ()=0

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

  Metoda wyswietlajaca dane.
- virtual void _Zwolnij ()=0

  Metoda zwalniajaca pamiec.

5.7.1 Opis szczegółowy

Modeluje pojecie Struktury danych, klasa bazowa dla Stosu,Kolejki i Listy,zarowno w implemenetacji wskaznikowej jak i tablicowej.

Definicja w linii 20 pliku Struktury.h.

5.7.2 Dokumentacja funkcji składowych

5.7.2.1 virtual void Struktury::_Pokaz ( ) [pure virtual]

Metoda wyswietlajaca dane.

Metoda ma za zadanie wyswietlic wszytskie dane nalezace do struktury

Implementowany w Stos, TabListaPod, Lista, Kolejka, ListaTab i StosTab.

5.7.2.2 virtual int Struktury::_Pop (unsigned int *Pozycja* = 1) [pure virtual]

Metoda usuwajaca element.

Metoda ma za zadanie usunac element i w zaleznosci od implementowanej struktury bedzie to usuwany element usuwany z poczatk,końca lub w przypadku listy z dowolnego jej miejsca

Parametry

in	*Pozycja*	- Numer elementu ,ktory zostanie dodany. Argument ma znaczenie tylko w przypadku listy i domyślnie jest ustawiony, tak aby element był dodawany zawsze na poczatku listy

Zwraca

Zwraca wartosc elementu z odpowiedniego dla wybranej struktury miejsca

Implementowany w Kolejka, ListaTab, Stos, StosTab, TabListaPod i Lista.

5.7.2.3 virtual void Struktury::_Push (int *k*, unsigned int *Pozycja* = 1) [pure virtual]

Metoda dodajaca kolejny element struktury.

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

Parametry

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

Implementowany w Kolejka, ListaTab, StosTab, TabListaPod, Stos i Lista.

5.7.2.4 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 Kolejka, ListaTab, StosTab, TabListaPod, Stos i Lista.

5.7.2.5 virtual void Struktury::_Zwolnij ( ) [pure virtual]

Metoda zwalniajaca pamiec.

Metoda ma za zadanie zwolnic pamiec uzywana przy zapelnienianiu danej struktry danymi

Implementowany w Kolejka, Stos, TabListaPod, Lista, ListaTab i StosTab.

Dokumentacja dla tej klasy została wygenerowana z pliku:

- Struktury.h

5.8 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.

5.8.1 Opis szczegółowy

Klasa modeluje pojecie Benchmarku przeznaczonego dla struktur danych przechowujace dane

Definicja w linii 18 pliku StrukturyBenchmark.h.

5.8.2 Dokumentacja konstruktora i destruktora

5.8.2.1 StrukturyBenchmark::StrukturyBenchmark ( ) [inline]

Konstruktor.

Konstruktor sprawia ze wskazniki nowo powstalego obiektu wskazuja na NULL

Definicja w linii 66 pliku StrukturyBenchmark.h.

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

Destruktor.

Definicja w linii 79 pliku StrukturyBenchmark.h.

5.8.3 Dokumentacja funkcji składowych

5.8.3.1 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 13 pliku StrukturyBenchmark.cpp.

Oto graf wywoływań tej funkcji:

5.8.3.2 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 20 pliku StrukturyBenchmark.cpp.

5.8.3.3 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 60 pliku StrukturyBenchmark.h.

Oto graf wywoływań tej funkcji:

5.8.3.4 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:

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

Metoda zwalniajaca zaalokowana przez struktury pamiec.

Metoda ma za zadanie oproznic zaladowane do struktury dane

Parametry

in	*n*	- Ilosc danych ktora zostanie zwolniona

Implementuje BenchmarkInterfejs.

5.8.4 Dokumentacja atrybutów składowych

5.8.4.1 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 26 pliku StrukturyBenchmark.h.

5.8.4.2 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 33 pliku StrukturyBenchmark.h.

Dokumentacja dla tej klasy została wygenerowana z plików:

- StrukturyBenchmark.h
- StrukturyBenchmark.cpp

5.9 Dokumentacja klasy StrukturyTablice

`#include <StrukturyTablice.h>`

Metody publiczne

- virtual void _Push (int k, unsigned int Pozycja=0)=0
- virtual int _Pop (unsigned int Pozycja=0)=0

- virtual void _Pokaz ()=0
- virtual unsigned int _Rozmiar ()=0

5.9.1 Opis szczegółowy

Definicja w linii 8 pliku StrukturyTablice.h.

5.9.2 Dokumentacja funkcji składowych

5.9.2.1 virtual void StrukturyTablice::_Pokaz ( ) [pure virtual]

5.9.2.2 virtual int StrukturyTablice::_Pop (unsigned int *Pozycja* = 0) [pure virtual]

5.9.2.3 virtual void StrukturyTablice::_Push (int *k*, unsigned int *Pozycja* = 0) [pure virtual]

5.9.2.4 virtual unsigned int StrukturyTablice::_Rozmiar ( ) [pure virtual]

Dokumentacja dla tej klasy została wygenerowana z pliku:

- StrukturyTablice.h

5.10 Dokumentacja klasy TabListaPod

`#include <TabListaPod.h>`

Diagram dziedziczenia dla TabListaPod

Diagram współpracy dla TabListaPod:

Metody publiczne

- TabListaPod (const TabListaPod &K)

  Konstruktor Kopiujacy.
- TabListaPod ()

  Konstruktor Podczas tworzenia obiektu tej klasy automatycznie alokowana jest tablica o rozmiarze 1 oraz ustawienie obecnej liczby elementow listy na 0.
- ~TabListaPod ()

  Destruktor obiektu.
- int _Pop (unsigned int Pozycja)

  Metoda scigajaca element z listy Metoda sciagnac element i w przypadku ,gdy rozmiar tablicy bedzie dwuktornie wiekszy od ilosci elementow, zostanie zalokowana nowa tablica o zmniejszonym dwukrotnie rozmiarze.
- void _Push (int k, unsigned int Pozycja)

  Dodaje element do listy.
- unsigned int _Rozmiar ()

  Metoda zwracajaca rozmiar Listy.
- void _Pokaz ()

  Metoda wypisujaca elemeny listy.
- void _Zwolnij ()
- void _XRozszerz (const double Ile, unsigned int Pozycja, int k=0)

Atrybuty prywatne

- int * _L

  Pole klasy ListaTab.
- unsigned int _RozmiarL

  Pole Klasy ListaTab.
- unsigned int _RozmiarT

  Pole Klasy ListaTab.

5.10.1 Opis szczegółowy

Definicja w linii 13 pliku TabListaPod.h.

5.10.2 Dokumentacja konstruktora i destruktora

5.10.2.1 TabListaPod::TabListaPod (const TabListaPod & *K*)

Konstruktor Kopiujacy.

Definicja w linii 8 pliku TabListaPod.cpp.

5.10.2.2 TabListaPod::TabListaPod ( )

Konstruktor Podczas tworzenia obiektu tej klasy automatycznie alokowana jest tablica o rozmiarze 1 oraz ustawienie obecnej liczby elementow listy na 0.

Definicja w linii 41 pliku TabListaPod.cpp.

5.10.2.3 TabListaPod::~TabListaPod ( ) [inline]

Destruktor obiektu.

Definicja w linii 49 pliku TabListaPod.h.

5.10.3 Dokumentacja funkcji składowych

5.10.3.1 void TabListaPod::_Pokaz ( ) [virtual]

Metoda wypisujaca elemeny listy.

Metoda ma za zadanie wypisac wszystkie elementy znajdujace sie obecnie na liscie danych

Implementuje Struktury.

Definicja w linii 48 pliku TabListaPod.cpp.

5.10.3.2 int TabListaPod::_Pop (unsigned int *Pozycja*) [virtual]

Metoda scigajaca element z listy Metoda sciagnac element i w przypadku ,gdy rozmiar tablicy bedzie dwuktornie wiekszy od ilosci elementow, zostanie zalokowana nowa tablica o zmniejszonym dwukrotnie rozmiarze.

Parametry

in	*Pozycja*	- numer elementu ktory zostanie zwrocony z listy

Zwraca

Zwraca wartosc elementu z zadanej pozycji

Implementuje Struktury.

Definicja w linii 59 pliku TabListaPod.cpp.

5.10.3.3 void TabListaPod::_Push (int *k,* unsigned int *Pozycja*) [virtual]

Dodaje element do listy.

Metoda ma za zadanie dodac element do listy. W przypaku, gdy skonczy sie miejsce w tablicy,zostaje wowczas alokowana nowa tablica o dwukrotnie wiekszym rozmiarze, w inny przypadku element zostaje dodany w miejscu zadanym przez uzytkownika poprzez argument metody

Parametry

in	*k*	- Wartosc jaka chcemy umiescic na liscie
in	*Pozycja*	- Pozycja na ktorej chcemy dodac element

Implementuje Struktury.

Definicja w linii 21 pliku TabListaPod.cpp.

5.10.3.4 unsigned int TabListaPod::_Rozmiar ( ) [inline],[virtual]

Metoda zwracajaca rozmiar Listy.

Metoda ma za zadanie zwrocic informacje o aktualnej ilosci elementow na liscie

Zwraca

Liczba elementow listy

Implementuje Struktury.

Definicja w linii 84 pliku TabListaPod.h.

5.10.3.5 void TabListaPod::_XRozszerz (const double *Ile,* unsigned int *Pozycja,* int *k* = 0)

Definicja w linii 85 pliku TabListaPod.cpp.

5.10.3.6 void TabListaPod::_Zwolnij ( ) [virtual]

Metoda zwalniajaca pamiec

Metoda ma za zadanie zwolnij pamiec zajmowana przez dane, dopoki ilosc elementow listy nie wynosi 0 wykonywana jest metoda _Pop, aby oproznic stos i zwolnic pamiec

Implementuje Struktury.

Definicja w linii 15 pliku TabListaPod.cpp.

5.10.4 Dokumentacja atrybutów składowych

5.10.4.1 int* TabListaPod::_L [private]

Pole klasy ListaTab.

Pole zawiera wskaznik na typ calkowity, sluzy do alokacji pamieci na dynamiczna tablice

Definicja w linii 21 pliku TabListaPod.h.

5.10.4.2 unsigned int TabListaPod::_RozmiarL [private]

Pole Klasy ListaTab.

Pole przechowuje informacje o ilosci obecnie znajdujacych sie elementow na liscie danych

Definicja w linii 27 pliku TabListaPod.h.

5.10.4.3 unsigned int TabListaPod::_RozmiarT [private]

Pole Klasy ListaTab.

Pole przechowuje informacje o obecnycm rozmiarze tablicy danych

Definicja w linii 33 pliku TabListaPod.h.

Dokumentacja dla tej klasy została wygenerowana z plików:

- TabListaPod.h
- TabListaPod.cpp

5.11 Dokumentacja struktury Stos::Wezel

Diagram współpracy dla Stos::Wezel:

Atrybuty publiczne

- Wezel * _Nast

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

  Pole Wezla Pole przechowuje wartoc typu calkowitego.

5.11.1 Opis szczegółowy

Klasy Stos Pole modelje pojecie wezla,bedacego podstawa dla implementacji struktury danych w rozwinieciu wskaznikowym

Definicja w linii 27 pliku Stos.h.

5.11.2 Dokumentacja atrybutów składowych

5.11.2.1 Wezel* Stos::Wezel::_Nast

Pole Wezla Pole bedace wskaznikiem na kolejny element stosu.

Definicja w linii 32 pliku Stos.h.

5.11.2.2 int Stos::Wezel::_Wartosc

Pole Wezla Pole przechowuje wartoc typu calkowitego.

Definicja w linii 36 pliku Stos.h.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- Stos.h

5.12 Dokumentacja struktury Lista::Wezel

Diagram współpracy dla Lista::Wezel:

Atrybuty publiczne

- Wezel * _Nast

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

  Pole Wezla Pole przechowuje wartoc typu calkowitego.

5.12.1 Opis szczegółowy

Klasy lista Pole modelje pojecie wezla,bedacego podstawa dla implementacji struktury danych w rozwinieciu wskaznikowym

Definicja w linii 19 pliku Lista.h.

5.12.2 Dokumentacja atrybutów składowych

5.12.2.1 Wezel* Lista::Wezel::_Nast

Pole Wezla Pole bedace wskaznikiem na kolejny element listy.

Definicja w linii 24 pliku Lista.h.

5.12.2.2 int Lista::Wezel::_Wartosc

Pole Wezla Pole przechowuje wartoc typu calkowitego.

Definicja w linii 28 pliku Lista.h.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- Lista.h

5.13 Dokumentacja struktury Kolejka::Wezel

Diagram współpracy dla Kolejka::Wezel:

Atrybuty publiczne

- Wezel * _Nast

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

  Pole Wezla Pole przechowuje wartoc typu calkowitego.

5.13.1 Opis szczegółowy

Klasy Kolejka Pole modelje pojecie wezla,bedacego podstawa dla implementacji struktury danych w rozwinieciu wskaznikowym

Definicja w linii 29 pliku Kolejka.h.

5.13.2 Dokumentacja atrybutów składowych

5.13.2.1 Wezel* Kolejka::Wezel::_Nast

Pole Wezla Pole bedace wskaznikiem na kolejny element Kolejki.

Definicja w linii 34 pliku Kolejka.h.

5.13.2.2 int Kolejka::Wezel::_Wartosc

Pole Wezla Pole przechowuje wartoc typu calkowitego.

Definicja w linii 38 pliku Kolejka.h.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- Kolejka.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.

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:

Komponenty

- class Kolejka

  Modeluje pojecie Kolejki.
- struct Kolejka::Wezel

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 "Struktury.h"
```

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

Komponenty

- class Lista
- struct Lista::Wezel

6.7 Dokumentacja pliku ListaTab.cpp

Zawiera definicje Metod klasy ListaTab.

```
#include "ListaTab.h"
```

Wykres zależności załączania dla ListaTab.cpp:

6.7.1 Opis szczegółowy

Zawiera definicje Metod klasy ListaTab.

Definicja w pliku ListaTab.cpp.

6.8 Dokumentacja pliku ListaTab.h

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

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

Komponenty

- class ListaTab

6.9 Dokumentacja pliku Main.cpp

funkcja glowna programu

```
#include "ListaTab.h"
#include "TabListaPod.h"
#include "Lista.h"
#include "Stos.h"
#include "Kolejka.h"
#include "StosTab.h"
#include "StrukturyBenchmark.h"
```

Wykres zależności załączania dla Main.cpp:

Definicje

- #define ILOSC_DANYCH 100000000

Funkcje

- int main (int argc, char ∗argv[])

6.9.1 Opis szczegółowy

funkcja glowna programu

Definicja w pliku Main.cpp.

6.9.2 Dokumentacja definicji

6.9.2.1 #define ILOSC_DANYCH 100000000

Definicja w linii 8 pliku Main.cpp.

6.9.3 Dokumentacja funkcji

6.9.3.1 int main (int *argc,* char * *argv[]*)

Definicja w linii 14 pliku Main.cpp.

Oto graf wywołań dla tej funkcji:

6.10 Dokumentacja pliku Stos.cpp

Definicje metod klasy Stos.

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

6.10.1 Opis szczegółowy

Definicje metod klasy Stos.

Definicja w pliku Stos.cpp.

6.11 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.
- struct Stos::Wezel

6.12 Dokumentacja pliku StosTab.cpp

Zawiera definicje Metod klasy ListaTab.

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

6.12.1 Opis szczegółowy

Zawiera definicje Metod klasy ListaTab.

Definicja w pliku StosTab.cpp.

6.13 Dokumentacja pliku StosTab.h

```
#include <cstdlib>
```

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

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

Komponenty

- class StosTab

6.14 Dokumentacja pliku strona-glowna.dox

6.15 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,zarowno w implemenetacji wskaznikowej jak i tablicowej.

6.16 Dokumentacja pliku StrukturyBenchmark.cpp

Definicje metod klasy StrukturyBenchmark.

```
#include "StrukturyBenchmark.h"
```

Wykres zależności załączania dla StrukturyBenchmark.cpp:

6.16.1 Opis szczegółowy

Definicje metod klasy StrukturyBenchmark.

Definicja w pliku StrukturyBenchmark.cpp.

6.17 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

6.18 Dokumentacja pliku StrukturyTablice.h

```
#include <iostream>
#include <cstring>
#include <sstream>
#include <fstream>
```
Wykres zależności załączania dla StrukturyTablice.h:

Komponenty

- class StrukturyTablice

6.19 Dokumentacja pliku TabListaPod.cpp

Definicje Metod Klasy TabListaPod.

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

6.19.1 Opis szczegółowy

Definicje Metod Klasy TabListaPod.

Definicja w pliku TabListaPod.cpp.

6.20 Dokumentacja pliku TabListaPod.h

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

Komponenty

- class TabListaPod

Skorowidz