Program wczytujacy dane

2.1

Generated by Doxygen 1.8.4

Tue Apr 1 2014 21:15:16

Contents

Chapter 1

Operacje na pliku tekstowym

Author

Radoslaw Chudy

Date

30.03.2014

Version

2.0

```
Program wykonuje wczytuje dane z pliku tekstowego do struktur danych.
```

1.1 Dzialanie:

Program wykonuje operacje wczytywania danych z pliku do odpowiednich struktur danych tzn kolejki i stosu. Ponadto mierzy czas wykonywania tych operacji. Wyniki sa zapisywane w pliku csv.

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

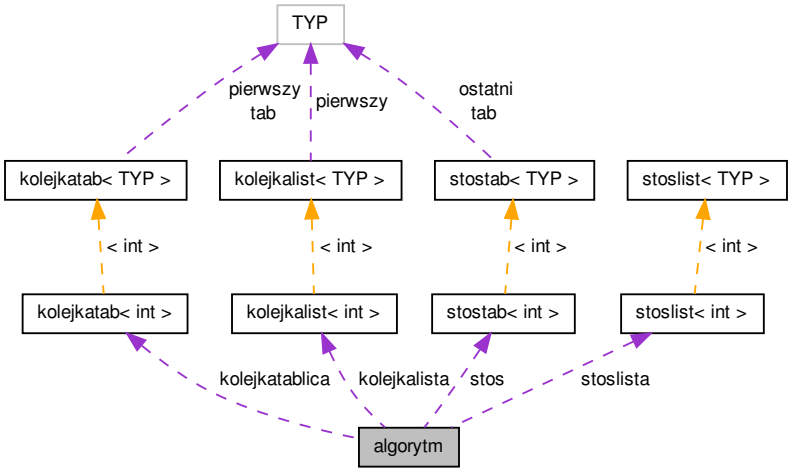
Class Documentation

4.1 algorytm Class Reference

Modeluje pojecie algorytmu.

`#include <algorytm.h>`

Collaboration diagram for algorytm:



Public Member Functions

- int * wczytaj_dane ()

  Metoda wczytujaca dane z pliku tesowego.
- void wczytaj_dane_stostab ()

  Metoda wczytujaca dane z pliku tekstowego do stosu implementowanego tablica.
- void wczytaj_dane_stoslista ()

  Metoda wczytujaca dane z pliku tekstowego do stosu implementowanego lista.
- void wczytaj_dane_kolejkatab ()

  Metoda wczytujaca dane z pliku tekstowego do kolejki implementowanej tablica.
- void wczytaj_dane_kolejkalist ()

Metoda wczytujaca dane z pliku tekstowego do kolejki implementowanej lista.

- void wlacz_zegar ()

  Metoda wlacz_zegar.
- void wylacz_zegar ()

  Metoda wylacz_zegar.
- int * wykonaj_obliczenia ()

  Metoda wykonaj obliczenia.
- int * wczytaj_dane_sprawdzajace ()

  Metoda wczytujaca dane sprawdzajace z pliku tesowego.
- bool sprawdz ()

  Metoda Sprawdz.
- void zamien_elementy (int *tablica, int pierwszy, int drugi)

  Metoda zamien elementy.
- void odwroc_tablice (int *tablica)

  Metoda odwroc tablice.
- int * dodaj_element (int *tablica, int element)

  Metoda dodaj elementy.
- int * dodaj_elementy (int *tab_pierwsza, int *tab_druga)

  Metoda dodaj elementy.
- int * operator= (int *tab_pierwsza)

  Przeciazenie operatora przypisania.
- int test (dane *Info)

  Metoda test.
- void test_wczytania (dane *Info)

  Metoda test_wczytania.

Public Attributes

- int powtorzenia

  Pole powtorzenia.
- double czas_nsec

  Pole czas_nsec.
- double czas_sec

  Pole czas_sec;.
- int elementy

  Pole elementy.
- int * tab_danych

  Wskaznik tab_danych.
- int * tab_obliczone

  Wskaznik tab_obliczone.
- int * tab_sprawdzajace

  Wskaznik tab_sprawdzajace.
- stostab< int > stos

  Obiekt stos klasy stostab.
- stoslist< int > stoslista

  Obiekt stoslista klasy stoslist.
- kolejkalist< int > kolejkalista

  Obiekt kolejkalista klasy kolejkalist.
- kolejkatab< int > kolejkatablica

  Obiekt kolejkatablica klasy kolejkatab.

4.1.1 Detailed Description

Klasa modeluje pojecie algorytmu w sklad ktorego wchodza pola odpowiadajace min. za ilosc wczytanych elementow, powtorzenia algorytmow. Wsrod metod klasy znajduja sie min. wczytanie danych, pobranie czasu, zamiana elementow tablicy itd.

Definition at line 28 of file algorytm.h.

4.1.2 Member Function Documentation

4.1.2.1 int * algorytm::dodaj_element (int * *tablica*, int *element*)

Metoda dodaj element dodaje do zadanej tablicy jeden element o zadanej wartosci na koniec tablicy przez co tablica zwieksza swoj rozmiar o 1. Metoda zwraca wskaznik na nowa tablice.

Parameters

in	*tablica*	- wskaznik na tablice do ktorej zostanie dodany element.
in	*element*	- zmienna typu int ktora zostanie dodana do podanej tablicy.

Returns

Zwraca wskaznik do nowej tablicy z dodanym elementem.

Definition at line 187 of file algorytm.cpp.

4.1.2.2 int * algorytm::dodaj_elementy (int * *tab_pierwsza*, int * *tab_druga*)

Metoda dodaje ze soba dwie tablice tworzac jeden element. Jako argumenty metoda przyjmuje wskazniki na tablice ktore maja byc ze soba dodane. Zwracany jest wskaznik na nowo stworzona tablice zlozona z elementow obu tablic.

Parameters

in	*tab_pierwsza*	- tablica od ktorej bedzie rozpoczynala sie nowo utworzona tablica.
in	*tab_druga*	- tablica ktora zostanie dopisana jako druga do nowo utworzonej tablicy.

Returns

Zwraca wskaznik na nowa tablice.

Definition at line 197 of file algorytm.cpp.

4.1.2.3 void algorytm::odwroc_tablice (int * *tablica*)

Metoda odwroc tablice wykonuje zamiane elementow tablicy w taki sposob iz pierwszy staje sie ostatni, a ostatni pierwszym. Jako argument przyjmuje wskaznik na tablice ktora chcemy odwrocic. Nie zwraca wartosci.

Parameters

in	*tablica*	- wskaznik na tablice wartosci int.

Definition at line 179 of file algorytm.cpp.

4.1.2.4 int * algorytm::operator= (int * *tab_pierwsza*)

Przeciazenie operatora przypisania pozwala na bezposrednie przypisanie dwoch tablic ze soba. Jako argument przyjmuje wskaznik na tablice ktora zostanie przypisana. Zwraca natomiast wskaznik na nowa tablice z przypisanymi elementami.

Parameters

in	*tab_pierwsza*	- wskaznik na tablice ktora bedzie przypisana do drugiej

Returns

Zwraca wskaznik na ta tablice.

Definition at line 209 of file algorytm.cpp.

4.1.2.5 bool algorytm::sprawdz ( )

Metoda ta sprawdza poprawnosci wykonania operacji na tablicy danych poprzez porownanie kazdego elementu tab_obliczone z odpowiadajacym jej elementem tab_sprawdzjace. W przypadku poprawnego wykonania poprawnej operacji na tab_danych i porownaniu z tab_sprawdzajace metoda zwraca wartosc true gdy jakis z elementow jest inny od odpowiednika to zwracana jest wartosc false.

Returns

Zwraca wartosc logiczna true/false w zaleznosci od rezultatu.

Definition at line 154 of file algorytm.cpp.

4.1.2.6 int algorytm::test (dane * *Info*)

Metoda test jak nazwa wskazuje dokonuje poprawnosci dzialania zalozen programu. Sklada sie z odpowiedniej sekkwencji wywolania funkcji. Otwierane sa tu pliki testowe i sprawdzajace. Do obiektu Info sa wysylane odpowiednie informacje. Na koniec wywolywane sa metody klasy dane dokonujace zapisu danych statystycznych do pliku csv. Jako argument przyjmuje wskaznik do obiektu klasy dane. Zwraca wartosc logiczna 0 po wykonaniu testu

Parameters

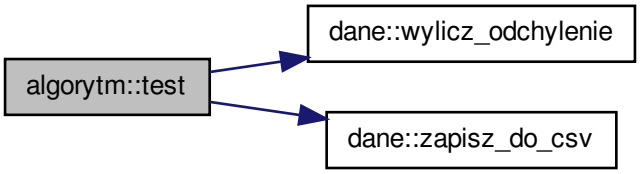
in	*Info*	- wskaznik na obiekt klasy dane do ktorego zapisane bd wyniki przeprowadzonego testu.

Returns

Zwraca wartosc logiczna typu int po wykonaniu testu.

Definition at line 218 of file algorytm.cpp.

Here is the call graph for this function:



4.1.2.7 void algorytm::test_wczytania (dane * *Info*)

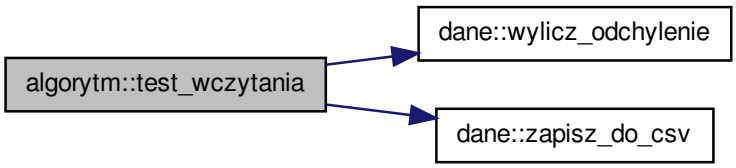
Metoda test_wczytania jest to metoda testujaca wczytywanie elementow z pliku oraz pomiar czasu tej operacji. Jako argument wejsciowy metoda przyjmuje wskaznik na obiekt typu dane. Nie zwracana jest zadna wartosc.

Parameters

in	*Info*	- wskaznik na obiekt klasy dane

Definition at line 236 of file algorytm.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

4.1.2.8 int * algorytm::wczytaj_dane ( )

Metoda ta wykonuje operacje wczytania pliku z danymi do tablicy tab_danych zawierajacej dane int. rozmiar tablicy definiowany jest przez liczbe elementow okreslona przez pierwsza wartosc wczytana z pliku. W przypadku niepowodzenia operacji otwarcia pliku uzytkownik informowany jest o tym odpowiednim komunikatem. Metoda zwraca wskaznik na tablice z wczytanymi przez nia danymi.

Returns

Zwraca wskaznik na tab_danych

Definition at line 30 of file algorytm.cpp.

4.1.2.9 void algorytm::wczytaj_dane_kolejkalist ( )

Metoda ta wykonuje operacje wczytywania pliku tekstowego z danymi do kolejki zaimplementowanej przy uzyciu szablonu listy z STL wraz z wykorzystaniem jej metod. Metoda nie przyjmuje zadnych argumentow. Nie zwraca tez zadnych wartosci , w przypadku niepoprawnego otwarcia pliku uzytkownik jest informowany o tym fakcie odpowiednim komunikatem.

Definition at line 100 of file algorytm.cpp.

4.1.2.10 void algorytm::wczytaj_dane_kolejkatab ( )

Metoda ta wykonuje operacje wczytywania pliku tekstowego z danymi do kolejki zaimlementowanej przy użyciu tablicy alokowanej dynamicznie. Funkcja nie zwraca zadnej wartosci. W przypaku niepowodzenia operacji otwarcia pliku uzytkownik informowany jest o tym odpowiednim komunikatem.

Definition at line 82 of file algorytm.cpp.

4.1.2.11 int * algorytm::wczytaj_dane_sprawdzajace ( )

Metoda ta wykonuje operacje wczytania pliku z danymi do tablicy tab_sprawdzajace zawierajacej dane int. rozmiar tablicy definiowany jest przez liczbe elementow okreslona przez pierwsza wartosc wczytana z pliku z danymi testowymi. W przypadku niepowodzenia operacji otwarcia pliku uzytkownik informowany jest o tym odpowiednim komunikatem. Metoda zwraca wskaznik na tablice z wczytanymi przez nia danymi sprawdzajacymi.

Returns

Zwraca wskaznik na tab_sprawdzajace.

Definition at line 137 of file algorytm.cpp.

4.1.2.12 void algorytm::wczytaj_dane_stoslista ( )

Metoda ta wykonuje operacje wczytywania pliku tekstowego z danymi do stosu zaimlementowanego przy użyciu listy z STL. Funkcja nie zwraca zadnej wartosci. W przypaku niepowodzenia operacji otwarcia pliku uzytkownik informowany jest o tym odpowiednim komunikatem.

Definition at line 65 of file algorytm.cpp.

4.1.2.13 void algorytm::wczytaj_dane_stostab ( )

Metoda ta wykonuje operacje wczytywania pliku tekstowego z danymi do stosu zaimlementowanego przy użyciu tablicy alokowanej dynamicznie. Funkcja nie zwraca zadnej wartosci. W przypaku niepowodzenia operacji otwarcia pliku uzytkownik informowany jest o tym odpowiednim komunikatem.

Definition at line 48 of file algorytm.cpp.

4.1.2.14 void algorytm::wlacz_zegar ( )

Metoda sluzy do wlaczenia zegara i wykorzystania w obliczeniach czasu w jakim wykonany zostal algorytm oraz kazde jego powtorzenie. Nie zwraca wartosci.

Definition at line 117 of file algorytm.cpp.

4.1.2.15 int * algorytm::wykonaj_obliczenia ( )

Metoda sluzy do wykonywania obliczen na tablicy znajdujacej sie w polu klasy Algorytm. Funkcja zwraca wskaznik na tablice z danymi po wykonaniu operacji na nich.

Returns

Wskaznik na tab_obliczone.

Definition at line 129 of file algorytm.cpp.

4.1.2.16 void algorytm::wylacz_zegar ( )

Metoda sluzy do wylaczenia zegara o wykonaniu sie danego algorytmu. Metoda nie zwraca zadnych wartosci.

Definition at line 123 of file algorytm.cpp.

4.1.2.17 void algorytm::zamien_elementy (int * *tablica,* int *pierwszy,* int *drugi*)

Metoda zamien elementy dokonuje zamiany elementow o zadanych parametrach. Jako parametry metoda przyjmuje wskaznik na tablice na ktorej chcemy wykonac operacje oraz dwa kolejne argumenty typu int, ktore podaja numer elementu ktore maja byc ze soba zamienione. Zadna wartosc nie jest zwracana.

Parameters

in	*tablica*	- wskaznik na tablice, w ktorej chcemy zamienic elementy.
in	*pierwszy*	- pierwszy z elementow tablicy ktore chcemy zamienic.
in	*drugi*	- drug z elementow tablicy ktore chcemy ze soba zamienic.

Definition at line 173 of file algorytm.cpp.

4.1.3 Member Data Documentation

4.1.3.1 double algorytm::czas_nsec

Pole odpowiada za przechowywanie nano czesci czasu wykonania operacji.

Definition at line 43 of file algorytm.h.

4.1.3.2 double algorytm::czas_sec

Pole czas_sec odpowiada za przechowywanie liczby sekund wykonania danego powtorzenia.

Definition at line 50 of file algorytm.h.

4.1.3.3 int algorytm::elementy

Pole to odpowiedzialne jest za przechowywanie informacji mowiacej o ilosci elementow jaka pozostanie wczytana do programu z pliku tekstowego.

Definition at line 58 of file algorytm.h.

4.1.3.4 kolejkalist<int> algorytm::kolejkalista

Obiekt kolejkalista jest szablonem klasy kolejkalist oznaczajacym kolejke zaimplementowana przy uzyciu szablonu listy z biblioteki STL.

Definition at line 108 of file algorytm.h.

4.1.3.5 kolejkatab<int> algorytm::kolejkatablica

Obiekt kolejkatablica jest kolejka zaimplementowana przy uzyciu szablonu klasy kolejkatab. Klasa ta wykorzystuje implementacje przy uzyciu tablicy alokowanej dynamicznie.

Definition at line 117 of file algorytm.h.

4.1.3.6 int algorytm::powtorzenia

Pole to odpowiedzialne jest za przechowywanie informacji mowiacej o ilosci powtorzen algorytmu.

Definition at line 36 of file algorytm.h.

4.1.3.7 stostab<int> algorytm::stos

Obiekt ten jest szablonem obiektu klasy stostab oznaczajacy stos zaimplementowany na bazie tablicy alokowanej dynamicznie

Definition at line 92 of file algorytm.h.

4.1.3.8 stoslist<int> algorytm::stoslista

Obiekt ten jest szablonem obiektu klasy stoslist oznaczajacym stos zaimplementowany na bazie szablonu listy z STL.

Definition at line 100 of file algorytm.h.

4.1.3.9 int* algorytm::tab_danych

Pole to odpowiedzialne jest za przechowywanie wskaznika do alokowanej dynamicznie tablicy danych zawierajacej elementy wczytane z pliku testowego.

Definition at line 67 of file algorytm.h.

4.1.3.10 int* algorytm::tab_obliczone

Pole to odpowiedzialne jest za przechowywanie wskaznika do alokowanej dynamicznie tablicy danych zawierajacej wartosci elementow z tablicy danych przez okreslona wartosc.

Definition at line 76 of file algorytm.h.

4.1.3.11 int* algorytm::tab_sprawdzajace

Pole to odpowiedzialne jest za przechowywanie wskaznika do alokowanej dynamicznie tablicy danych zawierajacej wartosci elementow z tablicy danych przez okreslona wartosc.

Definition at line 85 of file algorytm.h.

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

- algorytm.h
- algorytm.cpp

4.2 dane Class Reference

Modeluje pojecie dane.

`#include <dane.h>`

Public Member Functions

- dane (int powtorzenia)

  Konstruktor parametryczny.

- void wylicz_odchylenie ()

  Metoda wyliczajaca odchylenie standardowe.
- void zapisz_do_csv ()

  Metoda zapisujaca dane do pliku csv.

Public Attributes

- int powtorzenia

  Pole powtorzenia.
- double czas_operacji

  Pole czas_operacji.
- double odchylenie

  Pole odchylenie.
- double * tab_czasow

  *Pole *tab_czasow.*

4.2.1 Detailed Description

Klasa modeluje pojecie dane. W tej klasie znajduja sie informacje dotyczace podsumowania pracy algorytmu. Polami tej klasy sa zmienne odpowiadajace za min. ilosc powtorzen algorytmu. Metody tej klasy wyliczaja odchylenie standardowe oraz zapisuja dane podsumowujace w pliku csv.

Definition at line 26 of file dane.h.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 dane::dane (int *powtorzenia*)

Konstruktor jako parametr przyjmuje wartosc powtorzen, aby zaalokowac pamiec potrzebna do przechowywania czasu kazdego powtorzenia algorytmu.

Definition at line 20 of file dane.cpp.

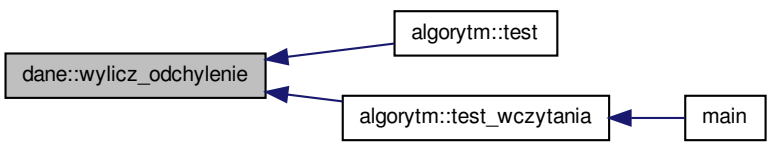
4.2.3 Member Function Documentation

4.2.3.1 void dane::wylicz_odchylenie ( )

Metoda ta wylicza wartosc odchylenia standardowego zestawu czasow zawartych w tablicy czasow tab_czasow. Metoda nie przyjmuje parametrow, ani nie zwraca zadnej wartosci.

Definition at line 23 of file dane.cpp.

Here is the caller graph for this function:

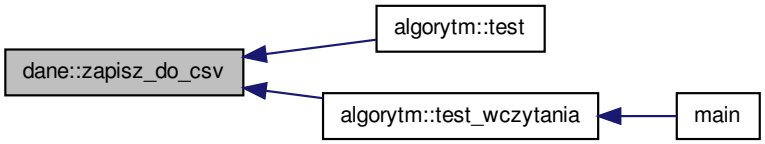


4.2.3.2 void dane::zapisz_do_csv ( )

Metoda ta zapisuje informacje dotyczace wykonania sie algorytmu w pliku formatu csv. W tej metodzie tworzony jest odpowiedni obiekt klasy fstream do ktorego zapisywane sa informacje odnosnie ilosci powtorzen, czasu operacji, czasu powtorzen oraz odchylenie standardowe czasow.

Definition at line 38 of file dane.cpp.

Here is the caller graph for this function:



4.2.4 Member Data Documentation

4.2.4.1 double dane::czas_operacji

Pole to odpowiedzialne jest za przechowywanie informacji mowiacej w jakim czasie wykonano wszystkie powtorzenia algorytmu. Zmienna jest typu double.

Definition at line 43 of file dane.h.

4.2.4.2 double dane::odchylenie

Pole to odpowiedzialne jest za przechowywanie informacji mowiacej ile wynosi odchylenie standardowe z czasow wykonania algorytmu. Zmienna jest typu double.

Definition at line 51 of file dane.h.

4.2.4.3 int dane::powtorzenia

Pole to odpowiedzialne jest za przechowywanie informacji mowiacej o ilosci powtorzen algorytmu. Zmienna jest typu int.

Definition at line 35 of file dane.h.

4.2.4.4 double* dane::tab_czasow

Pole to odpowiedzialne jest za przechowywanie czasow kazdego powtorzenia. Jest ono wskaznikiem na te tablice, ktora jest alokowana w sposob dynamiczny. Ilosc elementow okreslona jest przez ilosc powtorzen algorytmu.

Definition at line 60 of file dane.h.

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

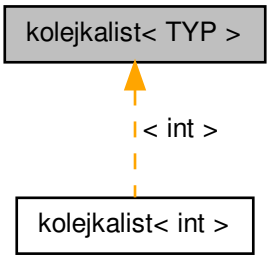
- dane.h
- dane.cpp

4.3 kolejkalist< TYP > Class Template Reference

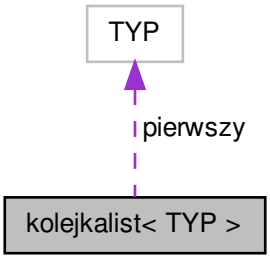
Definicja klasy kolejkalist.

`#include <kolejka.h>`

Inheritance diagram for kolejkalist< TYP >:



Collaboration diagram for kolejkalist< TYP >:



Public Member Functions

- kolejkalist ()

  Konstruktor klasy kolejkalist.
- int size ()

  Metoda size.
- bool empty ()

  Metoda empty.
- void enqueue (TYP element)

  Metoda enqueue.
- void dequeue ()

  Metoda dequeue.

Public Attributes

- list< TYP > lista

  Pole lista.
- TYP pierwszy

  Pole pierwszy.

4.3.1 Detailed Description

template<typename TYP>class kolejkalist< TYP >

Szablon klasy kolejkalist jest implementacja listy na bazie listy z kontenera STL. W sklad szablonu wchodzi szablon listy oraz pole klasy pierwszy typu TYP przechowujacy pierwszy element po wykonaniu operacji pobrania elemnetu.

Definition at line 29 of file kolejka.h.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 template<typename TYP> kolejkalist< TYP >::kolejkalist ( ) [inline]

W konstruktorze klasy inicjalizowana jest wartosc zmiennej pierwszy.

Definition at line 54 of file kolejka.h.

4.3.3 Member Function Documentation

4.3.3.1 template<typename TYP> void kolejkalist< TYP >::dequeue ( ) [inline]

Metoda ta zabiera pierwszy element znajdujacy sie w kolejce. Nie przyjmuje zadnych argumentow oraz nie zwraca zadnych wartosci. W przypadku braku elementow w kolejce uzytkownik jest informowany o tym fakcie odpowiednim komunikatem.

Definition at line 108 of file kolejka.h.

4.3.3.2 template<typename TYP> bool kolejkalist< TYP >::empty ( ) [inline]

Metoda ta sprawdza czy w kolejce znajduja sie jakiekolwiek elementy. Sprawdza ona warunek czy rozmiar kolejki zwracany w metodzie size jest rowny 0. Jeśli tak to zwraca wartosc logiczna true jesli nie to zwraca wartosc false. Metoda nie pobiera zadnych argumentow.

Returns

Zwraca wartosc logiczna w zaleznosci od wyniku operacji porownania

Definition at line 80 of file kolejka.h.

4.3.3.3 template<typename TYP> void kolejkalist< TYP >::enqueue (TYP *element*) [inline]

Metoda enqueue dodaje element na koniec kolejki. Wykorzystuje do tego metode z szablonu listy push_back ktora dodaje element na koniec listy. Metoda pobiera argument element typu TYP. Nie zwraca zadnych wartosci.

Parameters

in	*element*	- element ktory ma byc dolozony na koniec kolejki

Definition at line 97 of file kolejka.h.

4.3.3.4 template<typename TYP> int kolejkalist< TYP >::size ( ) [inline]

Metoda size podaje ilosc elementow znajdujacych sie w kolejce. Zwraca wynik funkci size z szablonu listy. Wartosc ta jest typu calkowitego.

Returns

Zwraca ilosc elementow w kolejce

Definition at line 66 of file kolejka.h.

4.3.4 Member Data Documentation

4.3.4.1 template<typename TYP> list<TYP> kolejkalist< TYP >::lista

Pole lista jest szablonem listy zmiennych typu TYP. W niej przechowywane sa elementy ktore dodawane sa do kolejki.

Definition at line 38 of file kolejka.h.

4.3.4.2 template<typename TYP> TYP kolejkalist< TYP >::pierwszy

Pole pierwszy jest zmienna typu takiego jaki zostanie podany przez uzytkownika w momencie tworzenia obiektu. Przechowywana w nim jest wartosc elementu ktory jako ostatni zostal pobrany z kolejki.

Definition at line 47 of file kolejka.h.

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

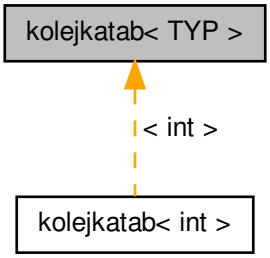
- kolejka.h

4.4 kolejkatab< TYP > Class Template Reference

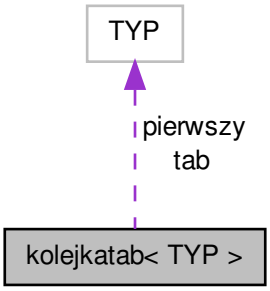
Szablon klasy kolejkatab.

`#include <kolejka.h>`

Inheritance diagram for kolejkatab< TYP >:



Collaboration diagram for kolejkatab< TYP >:



Public Member Functions

- kolejkatab ()
  Konstruktor klasy kolejkatab.
- TYP * tworz (int N)
  Metoda tworz.
- bool empty ()
  Metoda empty.
- int size ()
  Metoda size.
- void enqueue_dod (TYP element)
  Metoda enqueue_dod.
- void enqueue_pom (TYP element)
  Metoda enqueue_pom.
- void dequeue ()
  Metoda dequeue.
- void clear ()
  Metoda clear.

Public Attributes

- int rozmiar

  Pole rozmiar.
- TYP * tab

  Pole wskaznika na tablice.
- unsigned int ilosc

  Pole ilosc.
- TYP pierwszy

  Pole pierwszy.

4.4.1 Detailed Description

template<typename TYP>class kolejkatab< TYP >

Szablon klasy kolejkatab jest implementacja kolejki w oparciu o tablice alokowana dynamicznie. W szablonie znajduja sie wszystkie niezbedne metody do poprawnego funkcjonowania obiektow jako kolejek.

Definition at line 128 of file kolejka.h.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 template<typename TYP> kolejkatab< TYP >::kolejkatab ( ) [inline]

Konstruktor klasy kolejkatab. W nim przypisywana jest wartosc poczatkowa polom ilosc oraz rozmiar.

Definition at line 167 of file kolejka.h.

4.4.3 Member Function Documentation

4.4.3.1 template<typename TYP> void kolejkatab< TYP >::clear ( ) [inline]

Metoda wykonujaca operacje sciagniecia wszytskich elementow z kolejki. Zapobiega przed nadpisywaniem wartosci.

Definition at line 329 of file kolejka.h.

4.4.3.2 template<typename TYP> void kolejkatab< TYP >::dequeue ( ) [inline]

Metoda usuwajaca pierwszy element z kolejki. W przypadku gdy w kolejce nie ma elementow operacja usuniecia elementu nie zostanie wykonana. Uzytkownik jest o tym fakcie informowany odpowiednim komunikatem. Gdy kolejka wypelniona jest w co najmniej w 1/4 to rozmiar zaalokowanej tablicy zmniejszany jest o polowe. Metoda nie zwraca zadnych wartosci. Wartosc usunietego elementu przechowywana jest w polu pierwszy dopoki nie zostanie wykonana kolejna operacja usuniecia elementu z kolejki.

Definition at line 292 of file kolejka.h.

4.4.3.3 template<typename TYP> bool kolejkatab< TYP >::empty ( ) [inline]

Metoda empty sluzy do sprawdzenia czy w kolejce znajduja sie jakies elementy. Jesli wartosc zwracana przez metoda size jest inna od 0 wtedy funkcja zwraca wartosc false. W przypadku prawdziwosci warunku zwracana jest wartosc true.

Returns

Zwraca wartosc logiczna w zaleznosci od warunku petli

Definition at line 198 of file kolejka.h.

4.4.3.4 template<typename TYP> void kolejkatab< TYP >::enqueue_dod (TYP *element*) [inline]

Metoda enqueue_dod dokonuje dodania elementu na koniec kolejki. W przypadku kiedy tablica do ktorej wprowadzane sa wartosci jest zapelniona to jej rozmiar zwiekszany jest o 1. Jako argument metoda przyjmuje wartosc elementu typu TYP. Nie zwraca zadnych wartosci.

Parameters

in	*element*	- zmienna typu TYP dodawana na koniec kolejki

Definition at line 228 of file kolejka.h.

4.4.3.5 template<typename TYP> void kolejkatab< TYP >::enqueue_pom (TYP *element*) [inline]

Metoda enqueue_dod dokonuje dodania elementu na koniec kolejki. W przypadku kiedy tablica do ktorej wprowadzane sa wartosci jest zapelniona to jej rozmiar zwiekszany dwukrotnie. Jako argument metoda przyjmuje wartosc elementu typu TYP. Nie zwraca zadnych wartosci.

Parameters

in	*element*	- zmienna typu TYP dodawana na koniec kolejki

Definition at line 259 of file kolejka.h.

4.4.3.6 template<typename TYP> int kolejkatab< TYP >::size ( ) [inline]

Metoda size sluzy do podania ilosci elementow w kolejce. Zwraca aktualna wartosc zmiennej ilosc. Zwracana wartosc jest typu int.

Returns

Zwraca wartosc zmiennej ilosc.

Definition at line 213 of file kolejka.h.

4.4.3.7 template<typename TYP> TYP* kolejkatab< TYP >::tworz (int *N*) [inline]

Metoda tworz tworzy tablice dynamiczna o zadanym rozmiarze. Jako argument przyjmuje zmienna N typu int. Metoda zwraca wskaznik na tablice typow TYP.

Parameters

in	*N*	- wartosc rozmiaru zadeklarowanej tablicy

Returns

Wskaznik na tablice typow TYP

Definition at line 182 of file kolejka.h.

4.4.4 Member Data Documentation

4.4.4.1 template<typename TYP> unsigned int kolejkatab< TYP >::ilosc

Pole ilosc odpowiada za przechowywanie informacji ile elementow znajduje sie w kolejce. Sluzy do sprawdzenia wypelnienia tablicy.

Definition at line 152 of file kolejka.h.

4.4.4.2 template<typename TYP> TYP kolejkatab< TYP >::pierwszy

Pole pierwszy przechowuje wartosc pierwszego elementu znajdujacego sie w kolejce.

Definition at line 160 of file kolejka.h.

4.4.4.3 template<typename TYP> int kolejkatab< TYP >::rozmiar

Rozmiar jest zmienna typu int ktora przechowuje rozmiar zaalokowanej dynamicznie tablicy w ktorej znajduje sie kolejka.

Definition at line 136 of file kolejka.h.

4.4.4.4 template<typename TYP> TYP* kolejkatab< TYP >::tab

Pole to jest wskaznikiem na tablice tab ktora bedzie przechowywala elementy znajdujace sie w kolejce. Zainicjowanie tablicy dokonuje sie w w metodzie tworz po podaniu rozmiaru tablicy.

Definition at line 144 of file kolejka.h.

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

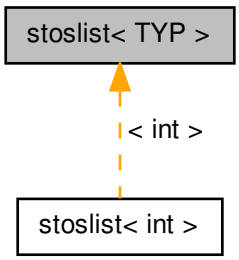
- kolejka.h

4.5 stoslist< TYP > Class Template Reference

Definicja szablonu stoslist.

`#include <stos.h>`

Inheritance diagram for stoslist< TYP >:



Public Member Functions

- int size ()

  Metoda size.
- bool empty ()

  Metoda empty.
- void push (TYP element)

  Metoda push.
- void pop ()

  Metoda pop.

Public Attributes

- list< TYP > lista

  Pole lista.
- TYP ostatni

  Pole ostatni.

4.5.1 Detailed Description

template<typename TYP>class stoslist< TYP >

Szablon stoslist jest implementacja stosu przy uzyciu szablonu listy dostepnego w STL list. W sklad szablonu wchodzi uzyty jako pole klasy szablon listy dzialajacy dla roznych typow danych oraz pole ostatni ktore przechowuje ostatni element znajdujacy sie na stosie.

Definition at line 27 of file stos.h.

4.5.2 Member Function Documentation

4.5.2.1 template<typename TYP> bool stoslist< TYP >::empty ( ) [inline]

Metoda empty sprawdza czy na stosie znajduja sie elementy. Jesli wynik wartosci zwracanej przez metode size jest inny od 0 to funkcja zwraca wartosc logiczna false. W przypadku gdy metoda size zwraca wartosc 0 to zwracana jest wartosc logiczna true.

Returns

Zwraca w zaleznosci od wyniku true badz false

Definition at line 66 of file stos.h.

4.5.2.2 template<typename TYP> void stoslist< TYP >::pop ( ) [inline]

Metoda pop odpowiada za zabranie elementu ze stosu (z konca listy). W tej metodzie wykorzystywana jest metoda pop_back ktora usuwa ostatni element z listy. Przed wykonaniem usuniecia elementu z listy do pola ostatni zapisywana jest wartosc ostatniego elementu znajdujacego sie w liscie. Przed wykonaniem jakiejkolwiek operacji sprawdzana jest liczba elementow na stosie. W przypadku gdy na stosie nie nie ma zadnego elementu to nie mozna wykonac operacji pobrania elementu ze stosu. W przypadku wywolania funkcji pop bez elementu na stosie. Uzytkownik informowany jest odpowiednim komunikatem o braku elementow na stosie. Metoda nie przyjmuje ani też nie zwraca zadnych elementow.

Definition at line 104 of file stos.h.

4.5.2.3 template<typename TYP> void stoslist< TYP >::push (TYP *element*) [inline]

Metoda push odpowiada za dodanie elementu do stosu (na koniec listy). Metoda bazuje na metodzie obiektu lista push_back, ktora dodaje element na koniec listy. Jako parametr wejsciowy metoda przyjmuje zmienna typu TYP. Metoda nie zwraca zadnej wartosci.

Parameters

in	*element*	- element ktory ma zostac dodany do struktury stosu

Definition at line 83 of file stos.h.

4.5.2.4 template<typename TYP> int stoslist< TYP >::size ( ) [inline]

Metoda size podaje ilosc elementow znajdujacych sie na stosie. Zwraca ona wynik metody z obiektu list.

Returns

Zwraca wartosc typu int wyniku metody lista.size()

Definition at line 52 of file stos.h.

4.5.3 Member Data Documentation

4.5.3.1 template<typename TYP> list<TYP> stoslist< TYP >::lista

Lista jest to miejsce przechowywania danych umieszczanych na stosie. Jest to obiekt typu list na bazie szablonu i dziala dla roznych typu danych.

Definition at line 36 of file stos.h.

4.5.3.2 template<typename TYP> TYP stoslist< TYP >::ostatni

Pole to odpowiada za przechowanie wartosci ostatniego elementu (pierwszy z gory) znajdujacego sie na stosie.

Definition at line 43 of file stos.h.

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

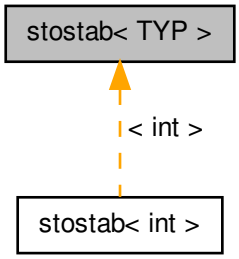
- stos.h

4.6 stostab< TYP > Class Template Reference

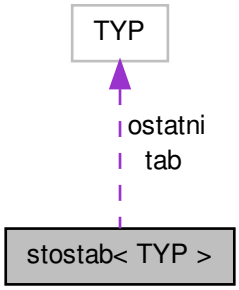
Definicja szablonu stostab.

`#include <stos.h>`

Inheritance diagram for stostab< TYP >:



Collaboration diagram for stostab< TYP >:



Public Member Functions

- stostab ()
  Kontruktor stostab.
- ~stostab ()
  Destruktor stostab.
- int * tworz (int N)
  Metoda tworz.
- void push_dod (TYP element)
  Metoda push_dod.
- void push_pom (TYP element)
  Metoda push_pom.
- void pop ()
  Metoda pop.
- int size ()
  Metoda size.
- bool empty ()

Metoda empty.

- void clear ()

  Metoda clear.

Public Attributes

- int rozmiar

  Pole rozmiar.

- unsigned int ilosc

  Pole ilosc.

- TYP * tab

  Pole tab.

- TYP ostatni

  Pole ostatni.

4.6.1 Detailed Description

template<typename TYP>class stostab< TYP >

Szablon stostab jest implementacja stosu przy uzyciu tablicy alokowanej dynamicznie. Zawiera on w sobie metody sprawdzajace ilosc oraz czy sa jakiekolwiek elementy na stosie. Ponadto zawiera operacje dodania elementu na stos (na dwa sposoby: przy zwiekszaniu rozmiaru tablicy o 1 oraz zwiekszaniu rozmiaru tablicy dwukrotnie) oraz operacje pobrania elementu ze stosu. W swojej strukturze zdefiniowana ma metode tworz ktora tworzy tablice o zadanym rozmiarze.

Definition at line 128 of file stos.h.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 template<typename TYP> stostab< TYP >::stostab ( ) [inline]

Konstruktor wywolywany przy kazdorazowym utworzeniu obiektu typu stostab. W nim przypisywane sa wartosci zmiennym ilosc oraz rozmiar ktore wynosza 0.

Definition at line 166 of file stos.h.

4.6.2.2 template<typename TYP> stostab< TYP >::~stostab ( ) [inline]

Destruktor obiektu stostab wywolywany przy zakonczeniu dzialania obiektu. W nim usuwana jest cala tablica dynamiczna tab.

Definition at line 177 of file stos.h.

4.6.3 Member Function Documentation

4.6.3.1 template<typename TYP> void stostab< TYP >::clear ( ) [inline]

Metoda clear sluzy do usuniecia wszystkich elementow ze stosu. Wykorzystywana jest w momencie kilkukrotnego mierzenia czasu operacji. Zapobiega przed zaalokowaniem za duzej pamieci. Nie zwraca zadnych wartosci. Nie przyjmuje tez argumentow.

Definition at line 344 of file stos.h.

4.6.3.2 template<typename TYP> bool stostab< TYP >::empty ( ) [inline]

Metoda empty sprawdza czy na stosie znajduja sie elementy. Petla warunkowa if sprawdza wartosc zwracana przez funkcje size. Jezeli jest ona rowna 0 to zwracana jest wartosc logiczna true. Natomiast kiedy size zwraca inna wartosc niz 0 to zwracana jest wartosc false. Metoda nie przyjmuje argumentow wejsciowych.

Returns

Zwraca wartosc logiczna typu bool w zaleznosci od rezultatu

Definition at line 330 of file stos.h.

4.6.3.3 template<typename TYP> void stostab< TYP >::pop ( ) [inline]

Metoda pop wykonuje operacje usuniecia elementu ze stosu. Nie przyjmuje argumentow wejsciowych. Przed wykonaniem operacji usuniecia elementu ze stosu sprawdzane jest czy na stosie znajduja sie elementy. Jesli tak to operacja moze zostac wykonana jezeli nie to uzytkownik jest informowany o tym fakcie odpowiednim komunikatem. Po wykonaniu usuniecia elementu z tablicy sprawdzana jest ilosc elementow w tablicy. Jezeli tablica jest zapelniona w 1/4 rozmiaru elementami to jej rozmiar jest zmniejszany o polowe. Jezeli nie jest spelniony ten warunek to tablica zachowuje swoj rozmiar. Metoda zwraca usuniety element ze stosu.

Returns

Zwraca ostatni element na stosie

Definition at line 283 of file stos.h.

4.6.3.4 template<typename TYP> void stostab< TYP >::push_dod (TYP *element*) [inline]

Metoda push_dod jest to metoda wykonujaca operacje dodania elementu na koniec tablicy. Opiera się ona na powiekszeniu tablicy za kazdym razem o jeden gdy ilosc elementow jest rowna rozmiarowi tablicy. Jako argument metoda przyjmuje element typu TYP. Nie zwraca zadnej wartosci. W funkcji przepisywany jest wskaznik z tablicy tworzonej tymczasowo tablicy do tablicy stosu. Po kazdorazowym dodaniu elementu na stos zmienna ilosc jest powiekszana o 1.

Parameters

in	*element*	- element ktory ma zostac polozony na stos

Definition at line 212 of file stos.h.

4.6.3.5 template<typename TYP> void stostab< TYP >::push_pom (TYP *element*) [inline]

Metoda push_pom jest to metoda wykonujaca operacje dodania elementu na koniec stosu(tablicy). Opiera się ona na powiekszaniu rozmiaru tablicy dwukrotnie zawsze kiedy ilosc elementow tablicy bedzie rowna rozmiarowi tablicy. Jako argument funkcja przyjmuje element typu TYP natomiast nie zwraca ona zadnej wartosci. W metodzie tworzona jest tymczasowa zmienna wymiar odpowiadajaca za zaalokowanie tymczasowej tablicy o rozmiarze 2 razy wiekszym w przypadku zapelnienia pierwotnej tablicy.

Parameters

in	*element*	- wartosc jaka ma zostac dodana na stos

Definition at line 246 of file stos.h.

4.6.3.6 template<typename TYP> int stostab< TYP >::size ( ) [inline]

Metoda size podaje ilosc elementow znajdujacych sie na stosie. Metoda ta zwraca wartosc typu int tzn aktualna wartosc pola ilosc klasy stostab.Nie przyjmuje zadnych wartosci.

Returns

Zwraca wartosc zmiennej ilosc

Definition at line 315 of file stos.h.

4.6.3.7 template<typename TYP> int* stostab< TYP >::tworz (int *N*) [inline]

Metoda tworz tworzy tablice dynamiczna o zadanym rozmiarze Metoda przyjmuje jako parametr wejsciowy zmienna typu int. Jest to wartosc przypisywana do zmiennej rozmiar ktora opisuje rozmiar inicjalizowanej tablicy. Metoda zwraca wskaznik na tablice dynamiczna tab.

Parameters

in	*N*	- wartosc rozmiaru alokowanej tablicy

Returns

Zwraca wskaznik na tablice tab

Definition at line 193 of file stos.h.

4.6.4 Member Data Documentation

4.6.4.1 template<typename TYP> unsigned int stostab< TYP >::ilosc

Pole ilosc odpowiada za przechowywanie informacji ile elementow znajduje sie aktualnie na stosie. Jest to zmienna typu int.

Definition at line 145 of file stos.h.

4.6.4.2 template<typename TYP> TYP stostab< TYP >::ostatni

Pole ostatni przechowuje wartosc ostatniego elementu pobranego ze stosu.

Definition at line 158 of file stos.h.

4.6.4.3 template<typename TYP> int stostab< TYP >::rozmiar

Pole rozmiar jest to zmienna typu int, ktora odpowiada za przechowywanie informacji o rozmiarze zaalokowanej dynamicznie tablicy.

Definition at line 137 of file stos.h.

4.6.4.4 template<typename TYP> TYP* stostab< TYP >::tab

Pole tab jest to wskaznik na tablice zmiennych typu TYP alokowanej dynamicznie w metodzie tworz.

Definition at line 152 of file stos.h.

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

- stos.h

Chapter 5

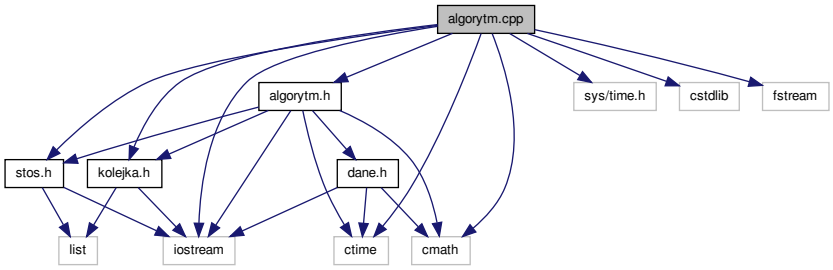
File Documentation

5.1 algorytm.cpp File Reference

Plik z definicjami metod dla klasy algorytm.

```
#include <iostream>
#include <ctime>
#include <sys/time.h>
#include <cstdlib>
#include <cmath>
#include <fstream>
#include "algorytm.h"
#include "stos.h"
#include "kolejka.h"
```

Include dependency graph for algorytm.cpp:



5.1.1 Detailed Description

Plik zawiera definicje metod klasy algorytm oraz przeciazenie operatora przypisania. Zdefiniowane sa tutaj metody odpowiadajace za wczytanie pliku z danymi oraz pliku z danymi sprawdzajacymi. Ponadto zdefiniowana sa metody pobrania czasu, wykonania obliczen algorytmu, sprawdzajaca poprawnosc obliczen oraz test algorytmu. w tym miejsu również znajduja sie definicje operacji odwracania elementow tablicy, zamiany danych elementow, dodanie elementu do tablicy, dodanie dwoch tablic. Zdefiniowane sa tez pliki testowe dla operacji wczytywania danych do struktur danych min kolejki i stosu.

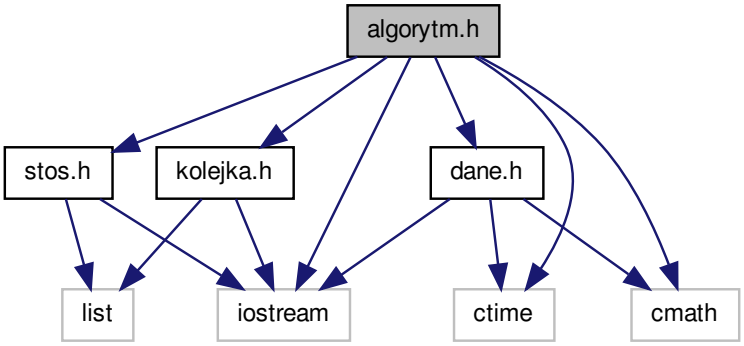
Definition in file algorytm.cpp.

5.2 algorytm.h File Reference

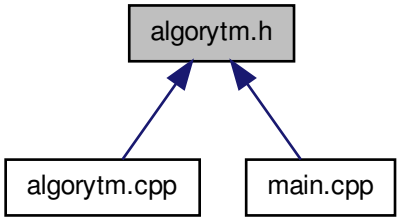
Definicja klasy algorytm.

```
#include <iostream>
#include <ctime>
#include <cmath>
#include "dane.h"
#include "stos.h"
#include "kolejka.h"
```

Include dependency graph for algorytm.h:



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



Classes

- class algorytm

  Modeluje pojecie algorytmu.

5.2.1 Detailed Description

Plik zawiera definicje klasy algorytm, ktora wykonuje niezbedne operacje na plikach tekstowych.

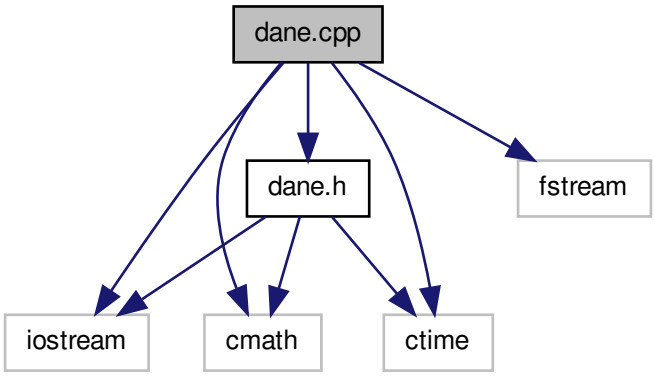
Definition in file algorytm.h.

5.3 dane.cpp File Reference

Plik z definicjami metod dla klasy dane.

```
#include <iostream>
#include <fstream>
#include <cmath>
#include <ctime>
#include "dane.h"
```

Include dependency graph for dane.cpp:



5.3.1 Detailed Description

Plik zawiera definicje metod klasy dane tj. wyliczania wartosci odchylenia standardowego oraz zapisywania danych do pliku formatu csv. Ponadto zawiera definicje konstruktora parametrycznego w ktorym alokowana jest tablica do ktorej beda wpisywane czasy poszczegolnych powtorzen.

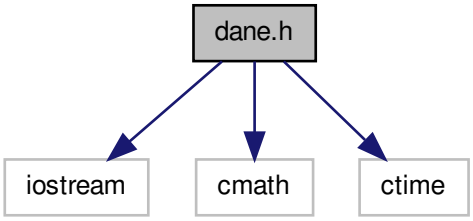
Definition in file dane.cpp.

5.4 dane.h File Reference

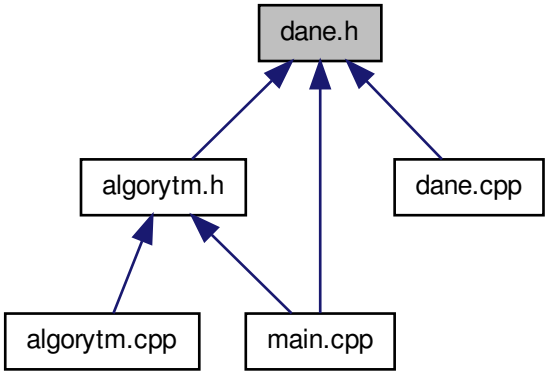
Definicja klasy dane.

```
#include <iostream>
#include <cmath>
#include <ctime>
```

Include dependency graph for dane.h:



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



Classes

- class dane

  Modeluje pojecie dane.

5.4.1 Detailed Description

Plik zawiera definicje klasy dane w ktorej znajduja sie informacje dotyczace wykonania algorytmu z powtorzeniami.

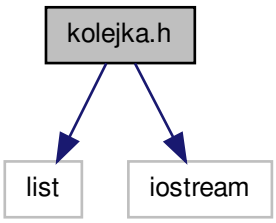
Definition in file dane.h.

5.5 kolejka.h File Reference

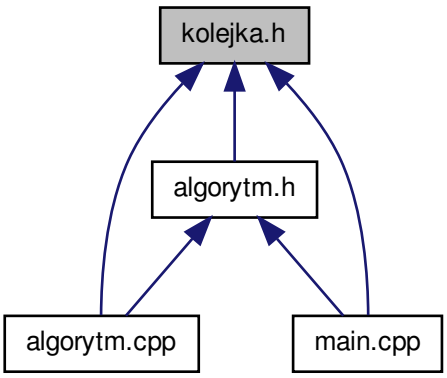
Modeluje pojecie kolejki.

```
#include <list>
#include <iostream>
```

Include dependency graph for kolejka.h:



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



Classes

- class kolejkalist< TYP >

  Definicja klasy kolejkalist.
- class kolejkatab< TYP >

  Szablon klasy kolejkatab.

5.5.1 Detailed Description

W pliku znajduja sie definicje klas tworzacych strukture danych jaka jest kolejka. Pierwsza z klas opiera sie na szablonie listy z kontenera STL druga zas opiera dzialanie na bazie tablicy alokowanej dynamicznie.

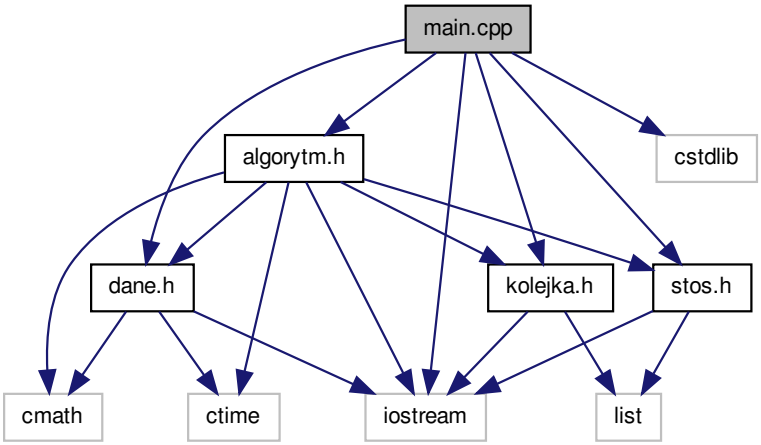
Definition in file kolejka.h.

5.6 main.cpp File Reference

Funkcja main.

```
#include <iostream>
#include <cstdlib>
#include "dane.h"
#include "algorytm.h"
#include "stos.h"
#include "kolejka.h"
```

Include dependency graph for main.cpp:



Functions

- int main ()

5.6.1 Detailed Description

Plik zawiera glowna funkcje main w ktorej wywolywana jest metoda test obiektu algorytm. Ponadto w niej wczytywana jest do pola obiektu klasy algorytm liczba powtorzen wykonania algorytmu. Po wykonaniu wszystkich operacji funkcja zwraca zero.

Returns

Zwraca wartosc logiczna 0.

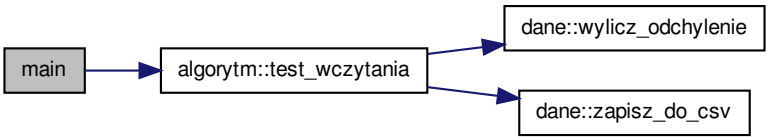
Definition in file main.cpp.

5.6.2 Function Documentation

5.6.2.1 int main ( )

Definition at line 22 of file main.cpp.

Here is the call graph for this function:

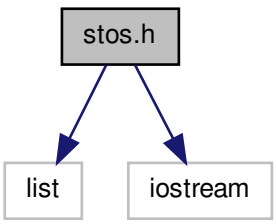


5.7 stos.h File Reference

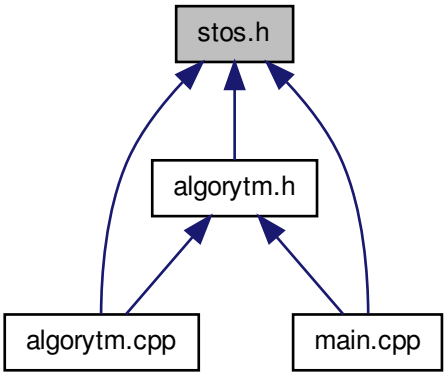
Modeluje pojecie stosu.

```
#include <list>
#include <iostream>
```

Include dependency graph for stos.h:



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



Classes

- class stoslist< TYP >

  Definicja szablonu stoslist.
- class stostab< TYP >

  Definicja szablonu stostab.

5.7.1 Detailed Description

Plik definuje pojecie Stosu w oparciu o dwie klasy wykorzystujace inne metody implementacji struktury. Klasa stoslist wykorzystuje szablon STL listy natomiast stostab implementuje tablice alokowana dynamicznie.

Definition in file stos.h.

5.8 strona.dox File Reference

Index