PAMSI - pwilkosz

1.0

Wygenerowano przez Doxygen 1.7.6.1

Sun Mar 16 2014 14:36:14

Spis treści

1 Indeks klas **1**

1.1 Hierarchia klas 1

2 Indeks klas **3**

2.1 Lista klas 3

3 Indeks plików **5**

3.1 Lista plików 5

4 Dokumentacja klas **7**

4.1 Dokumentacja klasy algorytm 7

4.1.1 Opis szczegółowy 9

4.1.2 Dokumentacja konstruktora i destruktora 9

4.1.2.1 algorytm 9

4.1.3 Dokumentacja funkcji składowych 10

4.1.3.1 ile_danych 10

4.1.3.2 jaki_czas 10

4.1.3.3 porownaj 10

4.1.3.4 przelicz 10

4.1.3.5 set_N 11

4.1.3.6 wczytaj 11

4.1.3.7 wczytaj_wzor 11

4.1.3.8 wlacz_zegar 12

4.1.3.9 wykonaj 12

4.1.3.10 wylacz_zegar 13

4.1.3.11 zapisz_do_csv 14

4.1.3.12 zapisz_do_gnuplot . . . 15
4.1.4 Dokumentacja atrybutów składowych . . . 15
4.1.4.1 czas . . . 16
4.1.4.2 dane . . . 16
4.1.4.3 dane_wz . . . 16
4.1.4.4 m . . . 16
4.1.4.5 n . . . 16
4.1.4.6 op . . . 16
4.2 Dokumentacja klasy kolejka_lista . . . 16
4.2.1 Opis szczegółowy . . . 18
4.2.2 Dokumentacja konstruktora i destruktora . . . 18
4.2.2.1 kolejka_lista . . . 18
4.2.3 Dokumentacja funkcji składowych . . . 18
4.2.3.1 przelicz . . . 18
4.2.4 Dokumentacja atrybutów składowych . . . 19
4.2.4.1 qu . . . 19
4.3 Dokumentacja klasy kolejka_tablica . . . 19
4.3.1 Opis szczegółowy . . . 21
4.3.2 Dokumentacja konstruktora i destruktora . . . 21
4.3.2.1 kolejka_tablica . . . 21
4.3.3 Dokumentacja funkcji składowych . . . 21
4.3.3.1 przelicz . . . 21
4.3.4 Dokumentacja atrybutów składowych . . . 22
4.3.4.1 qu . . . 22
4.4 Dokumentacja klasy mnozenie . . . 22
4.4.1 Opis szczegółowy . . . 24
4.4.2 Dokumentacja konstruktora i destruktora . . . 24
4.4.2.1 mnozenie . . . 24
4.4.3 Dokumentacja funkcji składowych . . . 24
4.4.3.1 przelicz . . . 24
4.5 Dokumentacja klasy operacje . . . 25
4.5.1 Opis szczegółowy . . . 25
4.5.2 Dokumentacja konstruktora i destruktora . . . 26
4.5.2.1 operacje . . . 26

4.5.2.2 operacje 26
4.5.3 Dokumentacja funkcji składowych 26
4.5.3.1 dodaj_element 26
4.5.3.2 dodaj_elementy 26
4.5.3.3 odwroc_tablice 26
4.5.3.4 operator= 27
4.5.3.5 operator== 27
4.5.3.6 zamien_elementy 27
4.5.4 Dokumentacja atrybutów składowych 27
4.5.4.1 n 27
4.5.4.2 tab 28
4.6 Dokumentacja szablonu klasy queue_array< TYP > 28
4.6.1 Opis szczegółowy 29
4.6.2 Dokumentacja konstruktora i destruktora 29
4.6.2.1 queue_array 29
4.6.2.2 queue_array 29
4.6.3 Dokumentacja funkcji składowych 29
4.6.3.1 clear 29
4.6.3.2 dequeue 30
4.6.3.3 enqueue 30
4.6.3.4 is_empty 30
4.6.3.5 size 31
4.6.4 Dokumentacja atrybutów składowych 31
4.6.4.1 f 31
4.6.4.2 q 31
4.6.4.3 s 31
4.6.4.4 sp 31
4.7 Dokumentacja szablonu klasy queue_list< TYP > 31
4.7.1 Opis szczegółowy 32
4.7.2 Dokumentacja funkcji składowych 32
4.7.2.1 clear 32
4.7.2.2 dequeue 33
4.7.2.3 enqueue 33
4.7.2.4 is_empty 33

4.7.2.5 size . . . 33
4.7.3 Dokumentacja atrybutów składowych . . . 33
4.7.3.1 q . . . 33
4.8 Dokumentacja szablonu klasy stack_array< TYP > . . . 34
4.8.1 Opis szczegółowy . . . 35
4.8.2 Dokumentacja konstruktora i destruktora . . . 35
4.8.2.1 stack_array . . . 35
4.8.2.2 stack_array . . . 35
4.8.3 Dokumentacja funkcji składowych . . . 35
4.8.3.1 clear . . . 35
4.8.3.2 is_empty . . . 36
4.8.3.3 pop . . . 36
4.8.3.4 push . . . 36
4.8.3.5 size . . . 37
4.8.4 Dokumentacja atrybutów składowych . . . 37
4.8.4.1 f . . . 37
4.8.4.2 s . . . 37
4.8.4.3 sp . . . 37
4.8.4.4 st . . . 37
4.9 Dokumentacja szablonu klasy stack_list< TYP > . . . 37
4.9.1 Opis szczegółowy . . . 38
4.9.2 Dokumentacja funkcji składowych . . . 38
4.9.2.1 clear . . . 38
4.9.2.2 is_empty . . . 38
4.9.2.3 pop . . . 39
4.9.2.4 push . . . 39
4.9.2.5 size . . . 39
4.9.3 Dokumentacja atrybutów składowych . . . 39
4.9.3.1 st . . . 39
4.10 Dokumentacja klasy stos_lista . . . 39
4.10.1 Opis szczegółowy . . . 41
4.10.2 Dokumentacja konstruktora i destruktora . . . 41
4.10.2.1 stos_lista . . . 41
4.10.3 Dokumentacja funkcji składowych . . . 41

4.10.3.1 przelicz . . . 41
4.10.4 Dokumentacja atrybutów składowych . . . 42
4.10.4.1 stos . . . 42
4.11 Dokumentacja klasy stos_tablica . . . 42
4.11.1 Opis szczegółowy . . . 44
4.11.2 Dokumentacja konstruktora i destruktora . . . 44
4.11.2.1 stos_tablica . . . 44
4.11.3 Dokumentacja funkcji składowych . . . 44
4.11.3.1 przelicz . . . 44
4.11.4 Dokumentacja atrybutów składowych . . . 45
4.11.4.1 stos . . . 45

5 Dokumentacja plików **47**

5.1 Dokumentacja pliku algorytm.cpp . . . 47
5.1.1 Opis szczegółowy . . . 47
5.2 Dokumentacja pliku algorytm.hh . . . 47
5.2.1 Opis szczegółowy . . . 49
5.3 Dokumentacja pliku kolejka.hh . . . 49
5.3.1 Opis szczegółowy . . . 50
5.4 Dokumentacja pliku main.cpp . . . 50
5.4.1 Opis szczegółowy . . . 51
5.4.2 Dokumentacja funkcji . . . 51
5.4.2.1 main . . . 51
5.5 Dokumentacja pliku operacje.cpp . . . 52
5.6 Dokumentacja pliku operacje.hh . . . 52
5.6.1 Dokumentacja definicji . . . 53
5.6.1.1 ROZMIAR . . . 53
5.7 Dokumentacja pliku statystyki.cpp . . . 54
5.7.1 Dokumentacja funkcji . . . 54
5.7.1.1 odchylenie_standardowe . . . 54
5.7.1.2 srednia . . . 55
5.8 Dokumentacja pliku statystyki.hh . . . 55
5.8.1 Opis szczegółowy . . . 56
5.8.2 Dokumentacja funkcji . . . 57

5.8.2.1 odchylenie_standardowe 57
5.8.2.2 srednia 57
5.9 Dokumentacja pliku stos.hh 58
5.9.1 Opis szczegółowy 59
5.9.2 Dokumentacja typów wyliczanych 60
5.9.2.1 flag 60
5.10 Dokumentacja pliku strona.dox 60

Rozdział 1

Indeks klas

1.1 Hierarchia klas

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

algorytm 7
- kolejka_lista 16
- kolejka_tablica 19
- mnozenie 22
- stos_lista 39
- stos_tablica 42

operacje 25
queue_array< TYP > 28
queue_list< TYP > 31
stack_array< TYP > 34
stack_list< TYP > 37

Rozdział 2

Indeks klas

2.1 Lista klas

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

algorytm
: Definicja klasy algorytm Jest to klasa bazowa, ktora ma za zadanie wczytac, przetworzyc i porownac dane z plikiem wzorcowym 7

kolejka_lista
: Klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury . 16

kolejka_tablica
: Klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury . 19

mnozenie
: Modeluje algorytm dokonujacy mnozenia kazdego elementu pliku wejsciowego przez 2 22

operacje
: Klasa modeluje tablice z danymi i metody sluzace do operacji na niej 25

queue_array< TYP >
: Modeluje kolejke w oparciu o tablice 28

queue_list< TYP >
: Modeluje kolejke oparta na liscie STL 31

stack_array< TYP >
: Modeluje stos w oparciu o tablice 34

stack_list< TYP >
: Modeluje stos oparty na liscie STL 37

stos_lista
: Klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury . 39

stos_tablica
: Klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury . 42

Rozdział 3

Indeks plików

3.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

algorytm.cpp
: Plik zawiera definicje metod klas zdefiniowanych w pliku algorytm.hh 47

algorytm.hh
: Definicja klas wykonujacych operacje na zestawie danych wejsciowych . . . 47

kolejka.hh
: Plik zawiera definicje klasy Kolejka Zaimplementowanej na 2 sposoby 1. Za pomocą listy. 2. Za pomocą tablicy a. kazdorazowo powiekszajacej swoj rozmiar b. powiekszajacej swoj rozmiar dwukrotnie, gdy kolejka sie przepelni . . . 49

main.cpp
: Plik glowny . . . 50

operacje.cpp . . . 52

operacje.hh . . . 52

statystyki.cpp . . . 54

statystyki.hh
: Plik zawiera dekalracje funkcji odpowiedzialnych za przeprowadznaie statystyk . . . 55

stos.hh
: Plik zawiera definicje klasy `Stos` Zaimplementowana na 2 sposoby 1. Za pomocą listy. 2. Za pomocą tablicy a. kazdorazowo powiekszajacej swoj rozmiar b. powiekszajacej swoj rozmiar dwukrotnie, gdy stos sie przepelni . . . 58

Rozdział 4

Dokumentacja klas

4.1 Dokumentacja klasy algorytm

Definicja klasy algorytm Jest to klasa bazowa, ktora ma za zadanie wczytac, przetworzyc i porownac dane z plikiem wzorcowym.

`#include <algorytm.hh>`

Diagram dziedziczenia dla algorytm

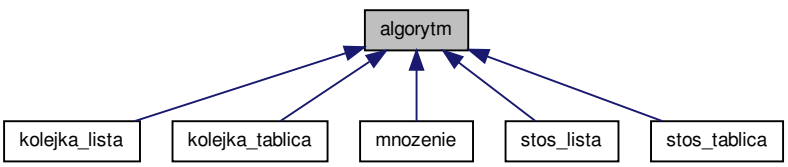
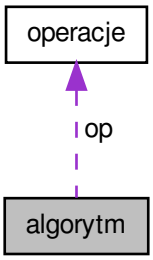


Diagram współpracy dla algorytm:



Metody publiczne

- algorytm (ifstream &plik1, ifstream &plik2, int N, int M)

  konstruktor kopiujacy - przekazuje informacje o nazwach plikow, ktore zapisywane sa do pol klasy
- void wykonaj (ofstream &out)

  funkcja dokonuje operacji na pliku wejsciowym, wywoluje metody odpowiedzialne za pomiar czasu oraz za porownanie wyniku operacji z plikiem wzorcowym
- bool wczytaj (ifstream &plik)

  Metoda wczytuje plik wejsciowy do tablicy `dane` *oraz do obiektu* `op` *klasy* `operacje`.
- void set_N (int wart)

  metoda ustawia wartosc n
- bool wczytaj_wzor (ifstream &plik)

  Metoda wczytuje plik wzorcowy do tablicy `dane_wz`.
- virtual float przelicz ()

  Metoda odpowiada za przetworzenie danych wejsciowych zgodnie z zadanym algorytmem.
- bool porownaj ()

  porownuje przetworzony dane z danymi wzorcowymi
- int ile_danych ()
- float ∗ jaki_czas ()
- float wlacz_zegar ()

  Metoda wlacza pomiar czasu poprzez wlaczenie funkcji `gettimeofday` *i przechowanie czasu w zmiennej* `start`.
- float wylacz_zegar ()

  Metoda wyacza pomiar czasu poprzez wlaczenie funkcji `gettimeofday` *i przechowanie czasu w zmiennej* `end`.

- void zapisz_do_csv (ofstream &out)

  Metoda zapisuje tablice `czas` do pliku `wyjscie.csv`.
- void zapisz_do_gnuplot (ofstream &out, float sr, float od)

  metoda zapisuje do pliku .csv parametry takie jak: srednia, ilosc liczb, odchylenie standardowe

Atrybuty publiczne

- float ∗ czas

  zawiera wyniki dzialania algorytmu

Atrybuty chronione

- float ∗ dane

  Tablica liczb wczytana z pliku.
- float ∗ dane_wz

  tablica liczb zawartych w pliku wzorcowym
- int n

  ilosc danych w pliku
- int m

  ilosc powtorzen
- operacje op

  klasa zawierajaca tablice i metody do operacji na niej

4.1.1 Opis szczegółowy

Definicja klasy algorytm Jest to klasa bazowa, ktora ma za zadanie wczytac, przetworzyc i porownac dane z plikiem wzorcowym.

Definicja w linii 33 pliku algorytm.hh.

4.1.2 Dokumentacja konstruktora i destruktora

4.1.2.1 algorytm::algorytm (ifstream & *plik1*, ifstream & *plik2*, int *N*, int *M*) [inline]

konstruktor kopiujacy - przekazuje informacje o nazwach plikow, ktore zapisywane sa do pol klasy

Parametry

in	*plik1*	- plik wejsciowy
in	*plik2*	- plik wzorcowy
in	*N*	- ilosc danych wejsciowych
in	*M*	- ilosc powtorzen

Definicja w linii 75 pliku algorytm.hh.

4.1.3 Dokumentacja funkcji składowych

4.1.3.1 int algorytm::ile_danych ( )

Zwraca

ilosc liczb wejsciowych

Definicja w linii 30 pliku algorytm.cpp.

4.1.3.2 float * algorytm::jaki_czas ( )

Zwraca

tablica `czas` z danymi pomiarowymi czasu wykonywania algorytmu

Definicja w linii 33 pliku algorytm.cpp.

4.1.3.3 bool algorytm::porownaj ( )

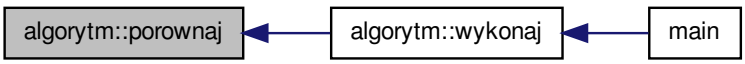
porownuje przetworzony dane z danymi wzorcowymi

Zwraca

true - gdy pliki zgodne false - w przeciwnym przypadku

Definicja w linii 82 pliku algorytm.cpp.

Oto graf wywoływań tej funkcji:



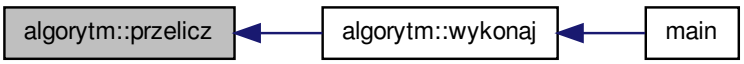
4.1.3.4 float algorytm::przelicz ( ) [virtual]

Metoda odpowiada za przetworzenie danych wejsciowych zgodnie z zadanym algorytmem.

Reimplementowana w kolejka_lista, kolejka_tablica, stos_lista, stos_tablica i mnozenie.

Definicja w linii 9 pliku algorytm.cpp.

Oto graf wywoływań tej funkcji:

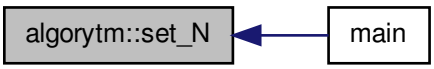


4.1.3.5 void algorytm::set_N (int *wart*) [inline]

metoda ustawia wartosc n

Definicja w linii 89 pliku algorytm.hh.

Oto graf wywoływań tej funkcji:



4.1.3.6 bool algorytm::wczytaj (ifstream & *plik*)

Metoda wczytuje plik wejsciowy do tablicy `dane` oraz do obiektu `op` klasy `operacje`.

Parametry

`in`	*plik*	- strumien pliku wejsciowego

Definicja w linii 10 pliku algorytm.cpp.

4.1.3.7 bool algorytm::wczytaj_wzor (ifstream & *plik*)

Metoda wczytuje plik wzorcowy do tablicy `dane_wz`.

Parametry

in	*plik*	- strumien pliku wejsciowego

Definicja w linii 21 pliku algorytm.cpp.

4.1.3.8 float algorytm::wlacz_zegar ( )

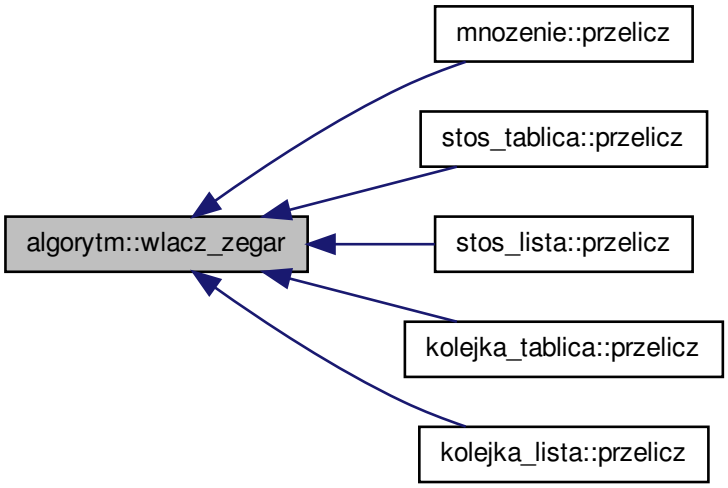
Metoda wlacza pomiar czasu poprzez wlaczenie funkcji `gettimeofday` i przechowanie czasu w zmiennej `start`.

Zwraca

start - zmienna pamietajaca czas poprzedzajacy wykonanie algorytmu

Definicja w linii 37 pliku algorytm.cpp.

Oto graf wywoływań tej funkcji:

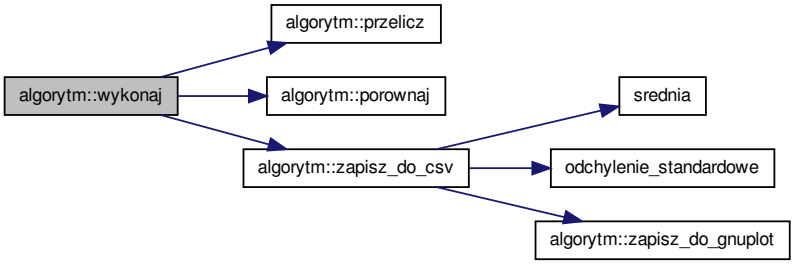


4.1.3.9 void algorytm::wykonaj (ofstream & *out*)

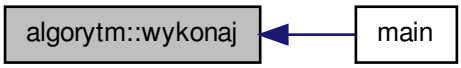
funkcja dokonuje operacji na pliku wejsciowym, wywoluje metody odpowiedzialne za pomiar czasu oraz za porownanie wyniku operacji z plikiem wzorcowym

Definicja w linii 66 pliku algorytm.cpp.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



4.1.3.10 float algorytm::wylacz_zegar ( )

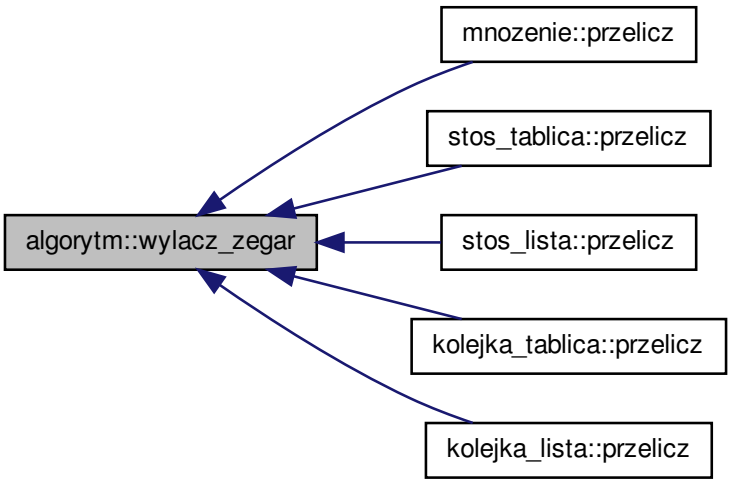
Metoda wyacza pomiar czasu poprzez wlaczenie funkcji `gettimeofday` i przechowanie czasu w zmiennej `end`.

Zwraca

end - zmienna pamietajaca czas poprzedzajacy wykonanie algorytmu

Definicja w linii 45 pliku algorytm.cpp.

Oto graf wywoływań tej funkcji:

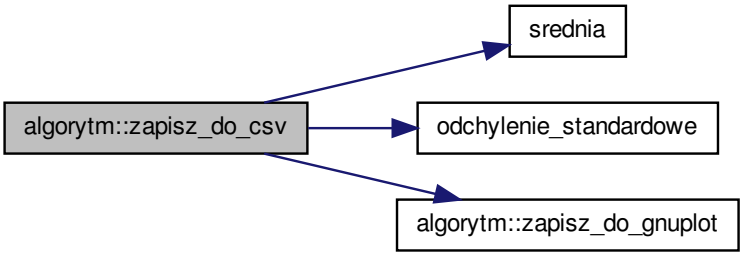


4.1.3.11 void algorytm::zapisz_do_csv (ofstream & *out*)

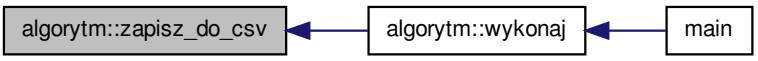
Metoda zapisuje tablice `czas` do pliku `wyjscie.csv`.

Definicja w linii 53 pliku algorytm.cpp.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

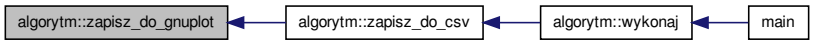


4.1.3.12 void algorytm::zapisz_do_gnuplot (ofstream & *out,* float *sr,* float *od*)

metoda zapisuje do pliku .csv parametry takie jak: srednia, ilosc liczb, odchylenie standardowe

Definicja w linii 89 pliku algorytm.cpp.

Oto graf wywoływań tej funkcji:



4.1.4 Dokumentacja atrybutów składowych

4.1.4.1 float* algorytm::czas

zawiera wyniki dzialania algorytmu

Definicja w linii 65 pliku algorytm.hh.

4.1.4.2 float* algorytm::dane [protected]

Tablica liczb wczytana z pliku.

Definicja w linii 41 pliku algorytm.hh.

4.1.4.3 float* algorytm::dane_wz [protected]

tablica liczb zawartych w pliku wzorcowym

Definicja w linii 46 pliku algorytm.hh.

4.1.4.4 int algorytm::m [protected]

ilosc powtorzen

Definicja w linii 56 pliku algorytm.hh.

4.1.4.5 int algorytm::n [protected]

ilosc danych w pliku

Definicja w linii 52 pliku algorytm.hh.

4.1.4.6 operacje algorytm::op [protected]

klasa zawierajaca tablice i metody do operacji na niej

Definicja w linii 60 pliku algorytm.hh.

Dokumentacja dla tej klasy została wygenerowana z plików:

- algorytm.hh
- algorytm.cpp

4.2 Dokumentacja klasy kolejka_lista

klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury

`#include <algorytm.hh>`

Diagram dziedziczenia dla kolejka_lista

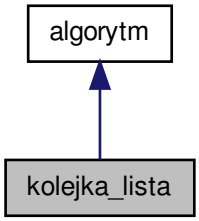
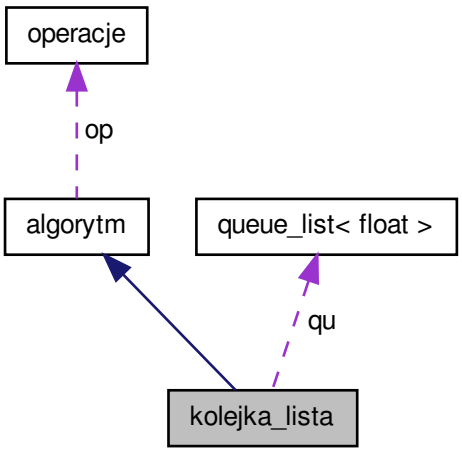


Diagram współpracy dla kolejka_lista:



Metody publiczne

- kolejka_lista (ifstream &plik1, ifstream &plik2, int N, int M)
- float przelicz ()

  Metoda odpowiada za przetworzenie danych wejsciowych zgodnie z zadanym algorytmem.

Atrybuty prywatne

- queue_list< float > qu

4.2.1 Opis szczegółowy

klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury

Definicja w linii 186 pliku algorytm.hh.

4.2.2 Dokumentacja konstruktora i destruktora

4.2.2.1 kolejka_lista::kolejka_lista (ifstream & *plik1,* ifstream & *plik2,* int *N,* int *M*) [inline]

Definicja w linii 189 pliku algorytm.hh.

4.2.3 Dokumentacja funkcji składowych

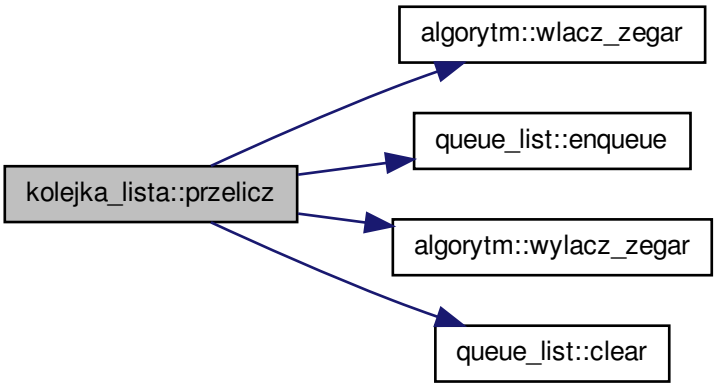
4.2.3.1 float kolejka_lista::przelicz ( ) [virtual]

Metoda odpowiada za przetworzenie danych wejsciowych zgodnie z zadanym algorytmem.

Reimplementowana z algorytm.

Definicja w linii 143 pliku algorytm.cpp.

Oto graf wywołań dla tej funkcji:



4.2.4 Dokumentacja atrybutów składowych

4.2.4.1 queue_list<float> kolejka_lista::qu [private]

Definicja w linii 187 pliku algorytm.hh.

Dokumentacja dla tej klasy została wygenerowana z plików:

- algorytm.hh
- algorytm.cpp

4.3 Dokumentacja klasy kolejka_tablica

klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury

`#include <algorytm.hh>`

Diagram dziedziczenia dla kolejka_tablica

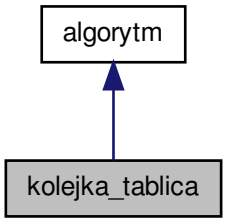
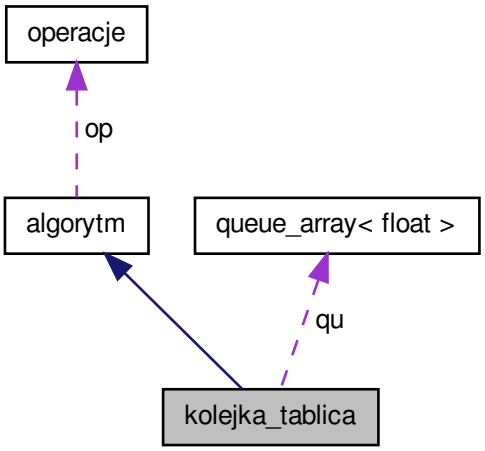


Diagram współpracy dla kolejka_tablica:



Metody publiczne

- kolejka_tablica (ifstream &plik1, ifstream &plik2, int N, int M, flag F)

  konstruktor - ustawia flage w zadany stan
- float przelicz ()

  Metoda odpowiada za przetworzenie danych wejsciowych zgodnie z zadanym algorytmem.

Atrybuty prywatne

- queue_array< float > qu

4.3.1 Opis szczegółowy

klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury

Definicja w linii 174 pliku algorytm.hh.

4.3.2 Dokumentacja konstruktora i destruktora

4.3.2.1 kolejka_tablica::kolejka_tablica (ifstream & *plik1,* ifstream & *plik2,* int *N,* int *M,* flag *F*) [inline]

konstruktor - ustawia flage w zadany stan

Definicja w linii 180 pliku algorytm.hh.

4.3.3 Dokumentacja funkcji składowych

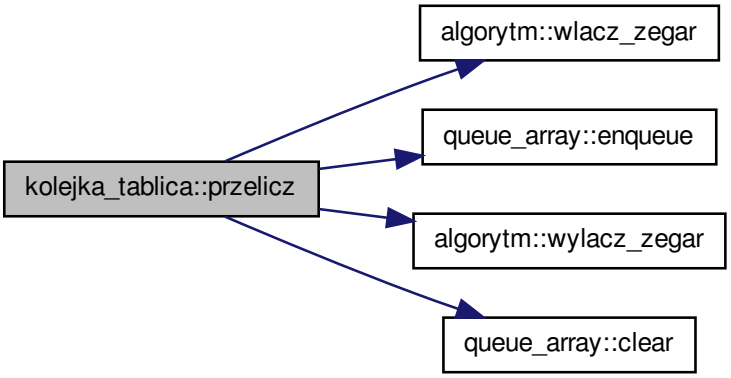
4.3.3.1 float kolejka_tablica::przelicz ( ) [virtual]

Metoda odpowiada za przetworzenie danych wejsciowych zgodnie z zadanym algorytmem.

Reimplementowana z algorytm.

Definicja w linii 130 pliku algorytm.cpp.

Oto graf wywołań dla tej funkcji:



4.3.4 Dokumentacja atrybutów składowych

4.3.4.1 queue_array<float> kolejka_tablica::qu [private]

Definicja w linii 175 pliku algorytm.hh.

Dokumentacja dla tej klasy została wygenerowana z plików:

- algorytm.hh
- algorytm.cpp

4.4 Dokumentacja klasy mnozenie

modeluje algorytm dokonujacy mnozenia kazdego elementu pliku wejsciowego przez 2

`#include <algorytm.hh>`

Diagram dziedziczenia dla mnozenie

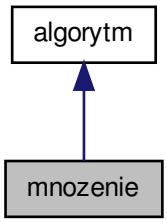
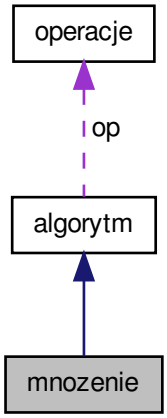


Diagram współpracy dla mnozenie:



Metody publiczne

- mnozenie (ifstream &plik1, ifstream &plik2, int N, int M)
- float przelicz ()

  wykonuje zalozony algorytm mnozenia elementow tablicy przez 2

4.4.1 Opis szczegółowy

modeluje algorytm dokonujacy mnozenia kazdego elementu pliku wejsciowego przez 2

Definicja w linii 134 pliku algorytm.hh.

4.4.2 Dokumentacja konstruktora i destruktora

4.4.2.1 mnozenie::mnozenie (ifstream & *plik1*, ifstream & *plik2*, int *N*, int *M*) [inline]

/brief konstruktor przekazuje do pol klasy informacje o nazwach pliku wejsciowego i wzorcowego

Parametry

in	*plik1*	- plik wejsciowy
in	*plik2*	- plik wzorcowy
in	*N*	- ilosc danych wejsciowych
in	*M*	- ilosc powtorzen

Definicja w linii 143 pliku algorytm.hh.

4.4.3 Dokumentacja funkcji składowych

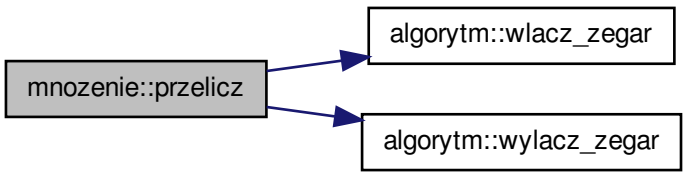
4.4.3.1 float mnozenie::przelicz ( ) [virtual]

wykonuje zalozony algorytm mnozenia elementow tablicy przez 2

Reimplementowana z algorytm.

Definicja w linii 95 pliku algorytm.cpp.

Oto graf wywołań dla tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z plików:

- algorytm.hh
- algorytm.cpp

4.5 Dokumentacja klasy operacje

Klasa modeluje tablice z danymi i metody sluzace do operacji na niej.

`#include <operacje.hh>`

Metody publiczne

- operacje ()

  konstruktor bezparametryczny
- operacje (int N)

  konstruktor parametryczny - alokuje pamiec w dynamicznej tablicy `tab`
- bool zamien_elementy (int i, int j)

  Metoda zamienia 2 elementy tablicy.
- void odwroc_tablice ()

  metoda odwraca wszystkie elementy tablicy
- void dodaj_element (float e)

  metoda dodaje element do tablicy, alokujac dodatkowa pamiec
- void dodaj_elementy (float ∗tab2, int rozm)

  metoda dodaje elementy do tablicy
- void operator= (float ∗tab1)

  Przeciazenie operatora przypisania; przypisuje elementy tablicy `tab1 do` *tablicy bedacej polem klasy.*
- bool operator== (float ∗tab1)

  Przeciazenie operatora porownania; metoda porownuje zawartosci dwoch tablic.

Atrybuty publiczne

- int n

  ilosc elementow w tablicy
- float ∗ tab

  tablica z liczbami

4.5.1 Opis szczegółowy

Klasa modeluje tablice z danymi i metody sluzace do operacji na niej.

Definicja w linii 9 pliku operacje.hh.

4.5.2 Dokumentacja konstruktora i destruktora

4.5.2.1 operacje::operacje ( )

konstruktor bezparametryczny

4.5.2.2 operacje::operacje (int *N*) [inline]

konstruktor parametryczny - alokuje pamiec w dynamicznej tablicy `tab`

Parametry

in	*N*	- ilosc elementow w tablicy; parametr przypisywany do pola `n` w klasie, oraz alokuje pamiec o takim wlasnie rozmiarze

Definicja w linii 26 pliku operacje.hh.

4.5.3 Dokumentacja funkcji składowych

4.5.3.1 void operacje::dodaj_element (float *e*)

metoda dodaje element do tablicy, alokujac dodatkowa pamiec

Parametry

in	*e*	- element, ktory nalezy dolaczyc do tablicy

Definicja w linii 27 pliku operacje.cpp.

4.5.3.2 void operacje::dodaj_elementy (float * *tab2,* int *rozm*)

metoda dodaje elementy do tablicy

Parametry

in	*tab2*	- tablica, ktora nalezy dolaczyc
in	*rozm*	- rozmiar tablicy tab2

Definicja w linii 46 pliku operacje.cpp.

4.5.3.3 void operacje::odwroc_tablice ( )

metoda odwraca wszystkie elementy tablicy

Definicja w linii 12 pliku operacje.cpp.

4.5.3.4 void operacje::operator= (float * *tab1*)

Przeciazenie operatora przypisania; przypisuje elementy tablicy `tab1` do tablicy bedacej polem klasy.

Parametry

in	*tab1*	- tablica, ktorej zawartosc przypisujemy

Definicja w linii 63 pliku operacje.cpp.

4.5.3.5 bool operacje::operator== (float * *tab1*)

Przeciazenie operatora porownania; metoda porownuje zawartosci dwoch tablic.

Parametry

in	*tab1*	- tablica, ktorej wartosci porownujemy

Zwraca

true - gdy zawartsoc tablic jest identyczna false - w przeciwnym przypadku

Definicja w linii 69 pliku operacje.cpp.

4.5.3.6 bool operacje::zamien_elementy (int *i*, int *j*)

Metoda zamienia 2 elementy tablicy.

Parametry

in	*i*	- element tablicy
in	*j*	- element tablicy

Zwraca

true - gdy elementy nie wykraczaja poza zakres tablicy false - w przeciwnym przypadku

Definicja w linii 3 pliku operacje.cpp.

4.5.4 Dokumentacja atrybutów składowych

4.5.4.1 int operacje::n

ilosc elementow w tablicy

Definicja w linii 14 pliku operacje.hh.

4.5.4.2 float* operacje::tab

tablica z liczbami

Definicja w linii 17 pliku operacje.hh.

Dokumentacja dla tej klasy została wygenerowana z plików:

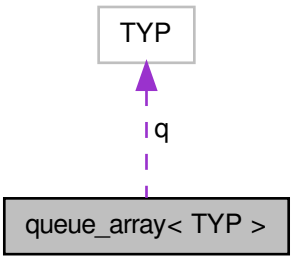
- operacje.hh
- operacje.cpp

4.6 Dokumentacja szablonu klasy queue_array< TYP >

Modeluje kolejke w oparciu o tablice.

`#include <kolejka.hh>`

Diagram współpracy dla queue_array< TYP >:



Metody publiczne

- queue_array ()

  konstruktor bezparametryczny
- queue_array (flag F)

  konstruktor parametryczny - ustawia flage na zadana pozycje
- int size ()
- bool is_empty ()
- void enqueue (TYP &element)

  Dodaje element na poczatek kolejki w zaleznosci od wybranego trybu powiekszania tablicy.
- TYP dequeue ()

usuwa element z konca kolejki

- void clear ()

  czysci kolejke

Atrybuty publiczne

- flag f

  flaga trybu zwiekszania pamieci , przyjmuje wartosc : plus1 - dla trybu kazdorazowego powiekszania pamieci x2 - dla trybu podwajania rozmiaru struktury

Atrybuty prywatne

- TYP * q
- int s
- int sp

4.6.1 Opis szczegółowy

template<typename TYP>class queue_array< TYP >

Modeluje kolejke w oparciu o tablice.

Definicja w linii 50 pliku kolejka.hh.

4.6.2 Dokumentacja konstruktora i destruktora

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

konstruktor bezparametryczny

Definicja w linii 63 pliku kolejka.hh.

4.6.2.2 template<typename TYP> queue_array< TYP >::queue_array (flag *F*) [inline]

konstruktor parametryczny - ustawia flage na zadana pozycje

Definicja w linii 65 pliku kolejka.hh.

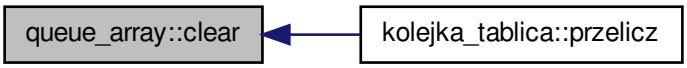
4.6.3 Dokumentacja funkcji składowych

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

czysci kolejke

Definicja w linii 170 pliku kolejka.hh.

Oto graf wywoływań tej funkcji:



4.6.3.2 template<typename TYP> TYP queue_array< TYP >::dequeue ( )

`[inline]`

usuwa element z konca kolejki

Definicja w linii 128 pliku kolejka.hh.

4.6.3.3 template<typename TYP> void queue_array< TYP >::enqueue (TYP & *element*)

`[inline]`

Dodaje element na poczatek kolejki w zaleznosci od wybranego trybu powiekszania tablicy.

Definicja w linii 82 pliku kolejka.hh.

Oto graf wywoływań tej funkcji:

4.6.3.4 template<typename TYP> bool queue_array< TYP >::is_empty ( )

`[inline]`

Zwraca

false - gdy kolejka nie jest pusta, true , gdy pusta

Definicja w linii 75 pliku kolejka.hh.

4.6.3.5 template<typename TYP> int queue_array< TYP >::size ( ) [inline]

Zwraca

rozmiar kolejki

Definicja w linii 70 pliku kolejka.hh.

4.6.4 Dokumentacja atrybutów składowych

4.6.4.1 template<typename TYP> flag queue_array< TYP >::f

flaga trybu zwiekszania pamieci , przyjmuje wartosc : plus1 - dla trybu kazdorazowego powiekszania pamieci x2 - dla trybu podwajania rozmiaru struktury

Definicja w linii 59 pliku kolejka.hh.

4.6.4.2 template<typename TYP> TYP* queue_array< TYP >::q [private]

Definicja w linii 51 pliku kolejka.hh.

4.6.4.3 template<typename TYP> int queue_array< TYP >::s [private]

Definicja w linii 52 pliku kolejka.hh.

4.6.4.4 template<typename TYP> int queue_array< TYP >::sp [private]

Definicja w linii 52 pliku kolejka.hh.

Dokumentacja dla tej klasy została wygenerowana z pliku:

- kolejka.hh

4.7 Dokumentacja szablonu klasy queue_list< TYP >

Modeluje kolejke oparta na liscie STL.

`#include <kolejka.hh>`

Metody publiczne

- bool is_empty ()
- int size ()
- void enqueue (TYP &element)

  dodaje element
- TYP dequeue ()

  usuwa element
- void clear ()

  czysci stos

Atrybuty prywatne

- list< TYP > q

4.7.1 Opis szczegółowy

template<typename TYP>class queue_list< TYP >

Modeluje kolejke oparta na liscie STL.

Definicja w linii 19 pliku kolejka.hh.

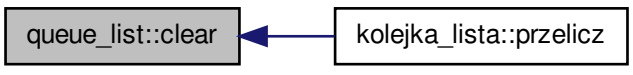
4.7.2 Dokumentacja funkcji składowych

4.7.2.1 template<typename TYP> void queue_list< TYP >::clear ( ) [inline]

czysci stos

Definicja w linii 41 pliku kolejka.hh.

Oto graf wywoływań tej funkcji:



4.7.2.2 template<typename TYP> TYP queue_list< TYP >::dequeue ( ) [inline]

usuwa element

Definicja w linii 35 pliku kolejka.hh.

4.7.2.3 template<typename TYP> void queue_list< TYP >::enqueue (TYP & *element*) [inline]

dodaje element

Definicja w linii 33 pliku kolejka.hh.

Oto graf wywoływań tej funkcji:

4.7.2.4 template<typename TYP> bool queue_list< TYP >::is_empty ( ) [inline]

Zwraca

false - gdy kolejka nie jest pusta, true , gdy pusta

Definicja w linii 26 pliku kolejka.hh.

4.7.2.5 template<typename TYP> int queue_list< TYP >::size ( ) [inline]

Zwraca

rozmiar kolejki

Definicja w linii 31 pliku kolejka.hh.

4.7.3 Dokumentacja atrybutów składowych

4.7.3.1 template<typename TYP> list<TYP> queue_list< TYP >::q [private]

Definicja w linii 20 pliku kolejka.hh.

Dokumentacja dla tej klasy została wygenerowana z pliku:

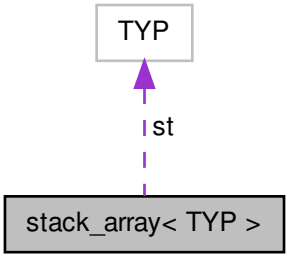
- kolejka.hh

4.8 Dokumentacja szablonu klasy stack_array< TYP >

Modeluje stos w oparciu o tablice.

`#include <stos.hh>`

Diagram współpracy dla stack_array< TYP >:



Metody publiczne

- stack_array ()

  konstruktor bezparametryczny
- stack_array (flag F)

  konstruktor parametryczny - ustawia flage na zadana pozycje
- bool is_empty ()
- int size ()
- void push (TYP &element)

  Dodaje element na wierzch stosu w zaleznosci od wybranego trybu powiekszania tablicy.
- TYP pop ()

  zdejmuje element ze stosu
- void clear ()

  czysci stos

Atrybuty publiczne

- flag f

flaga trybu zwiekszania pamieci , przyjmuje wartosc :
plus1 - dla trybu kazdorazowego powiekszania pamieci
x2 - dla trybu podwajania rozmiaru struktury

Atrybuty prywatne

- TYP * st
- int s
- int sp

4.8.1 Opis szczegółowy

template<typename TYP>class stack_array< TYP >

Modeluje stos w oparciu o tablice.

Definicja w linii 59 pliku stos.hh.

4.8.2 Dokumentacja konstruktora i destruktora

4.8.2.1 template<typename TYP> stack_array< TYP >::stack_array ( ) [inline]

konstruktor bezparametryczny

Definicja w linii 72 pliku stos.hh.

4.8.2.2 template<typename TYP> stack_array< TYP >::stack_array (flag *F*) [inline]

konstruktor parametryczny - ustawia flage na zadana pozycje

Definicja w linii 74 pliku stos.hh.

4.8.3 Dokumentacja funkcji składowych

4.8.3.1 template<typename TYP> void stack_array< TYP >::clear ( ) [inline]

czysci stos

Definicja w linii 178 pliku stos.hh.

Oto graf wywoływań tej funkcji:

4.8.3.2 template<typename TYP> bool stack_array< TYP >::is_empty ( ) [inline]

Zwraca

false - gdy stos nie jest pusty, true , gdy pusty

Definicja w linii 79 pliku stos.hh.

4.8.3.3 template<typename TYP> TYP stack_array< TYP >::pop ( ) [inline]

zdejmuje element ze stosu

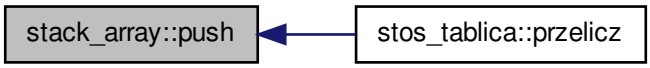
Definicja w linii 135 pliku stos.hh.

4.8.3.4 template<typename TYP> void stack_array< TYP >::push (TYP & *element*) [inline]

Dodaje element na wierzch stosu w zaleznosci od wybranego trybu powiekszania tablicy.

Definicja w linii 91 pliku stos.hh.

Oto graf wywoływań tej funkcji:



4.8.3.5 template<typename TYP> int stack_array< TYP >::size ( ) [inline]

Zwraca

rozmiar ztosu

Definicja w linii 87 pliku stos.hh.

4.8.4 Dokumentacja atrybutów składowych

4.8.4.1 template<typename TYP> flag stack_array< TYP >::f

flaga trybu zwiekszania pamieci , przyjmuje wartosc :

plus1 - dla trybu kazdorazowego powiekszania pamieci

x2 - dla trybu podwajania rozmiaru struktury

Definicja w linii 68 pliku stos.hh.

4.8.4.2 template<typename TYP> int stack_array< TYP >::s [private]

Definicja w linii 61 pliku stos.hh.

4.8.4.3 template<typename TYP> int stack_array< TYP >::sp [private]

Definicja w linii 61 pliku stos.hh.

4.8.4.4 template<typename TYP> TYP* stack_array< TYP >::st [private]

Definicja w linii 60 pliku stos.hh.

Dokumentacja dla tej klasy została wygenerowana z pliku:

- stos.hh

4.9 Dokumentacja szablonu klasy stack_list< TYP >

Modeluje stos oparty na liscie STL.

`#include <stos.hh>`

Metody publiczne

- bool is_empty ()
- int size ()

- void push (TYP &element)

  Dodaje element na wierzch stosu.
- TYP pop ()

  zdejmuje element z wierzchu stosu
- void clear ()

  czysci stos

Atrybuty prywatne

- list< TYP > st

4.9.1 Opis szczegółowy

template<typename TYP>class stack_list< TYP >

Modeluje stos oparty na liscie STL.

Definicja w linii 22 pliku stos.hh.

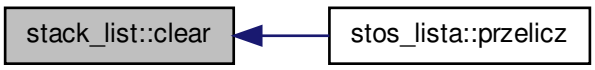
4.9.2 Dokumentacja funkcji składowych

4.9.2.1 template<typename TYP> void stack_list< TYP >::clear ( ) [inline]

czysci stos

Definicja w linii 50 pliku stos.hh.

Oto graf wywoływań tej funkcji:



4.9.2.2 template<typename TYP> bool stack_list< TYP >::is_empty ( ) [inline]

Zwraca

false - gdy stos nie jest pusty, true , gdy pusty

Definicja w linii 29 pliku stos.hh.

4.9.2.3 template<typename TYP> TYP stack_list< TYP >::pop ( ) [inline]

zdejmuje element z wierzchu stosu

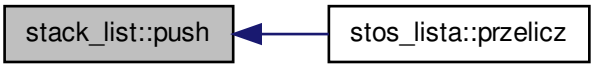
Definicja w linii 42 pliku stos.hh.

4.9.2.4 template<typename TYP> void stack_list< TYP >::push (TYP & *element*) [inline]

Dodaje element na wierzch stosu.

Definicja w linii 38 pliku stos.hh.

Oto graf wywoływań tej funkcji:



4.9.2.5 template<typename TYP> int stack_list< TYP >::size ( ) [inline]

Zwraca

rozmiar ztosu

Definicja w linii 34 pliku stos.hh.

4.9.3 Dokumentacja atrybutów składowych

4.9.3.1 template<typename TYP> list<TYP> stack_list< TYP >::st [private]

Definicja w linii 23 pliku stos.hh.

Dokumentacja dla tej klasy została wygenerowana z pliku:

- stos.hh

4.10 Dokumentacja klasy stos_lista

klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury

`#include <algorytm.hh>`

Diagram dziedziczenia dla stos_lista

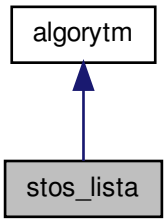
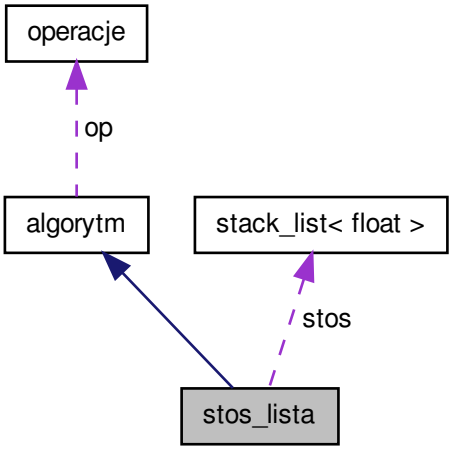


Diagram współpracy dla stos_lista:



Metody publiczne

- stos_lista (ifstream &plik1, ifstream &plik2, int N, int M)
- float przelicz ()

Metoda odpowiada za przetworzenie danych wejsciowych zgodnie z zadanym algorytmem.

Atrybuty prywatne

- stack_list< float > stos

4.10.1 Opis szczegółowy

klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury

Definicja w linii 165 pliku algorytm.hh.

4.10.2 Dokumentacja konstruktora i destruktora

4.10.2.1 stos_lista::stos_lista (ifstream & *plik1,* ifstream & *plik2,* int *N,* int *M*) [inline]

Definicja w linii 168 pliku algorytm.hh.

4.10.3 Dokumentacja funkcji składowych

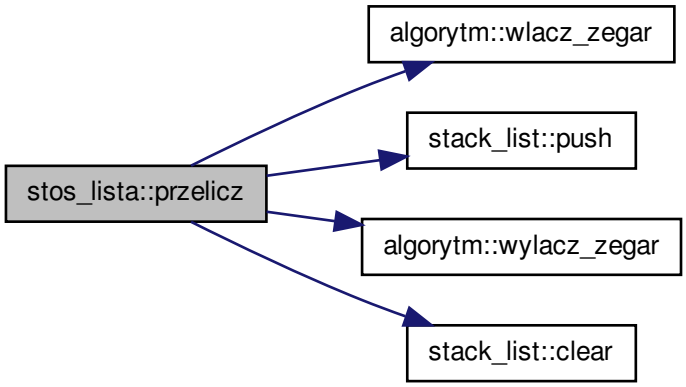
4.10.3.1 float stos_lista::przelicz ( ) [virtual]

Metoda odpowiada za przetworzenie danych wejsciowych zgodnie z zadanym algorytmem.

Reimplementowana z algorytm.

Definicja w linii 118 pliku algorytm.cpp.

Oto graf wywołań dla tej funkcji:



4.10.4 Dokumentacja atrybutów składowych

4.10.4.1 stack_list<float> stos_lista::stos [private]

Definicja w linii 166 pliku algorytm.hh.

Dokumentacja dla tej klasy została wygenerowana z plików:

- algorytm.hh
- algorytm.cpp

4.11 Dokumentacja klasy stos_tablica

klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury

`#include <algorytm.hh>`

Diagram dziedziczenia dla stos_tablica

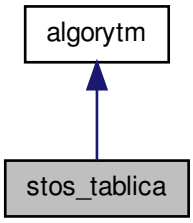
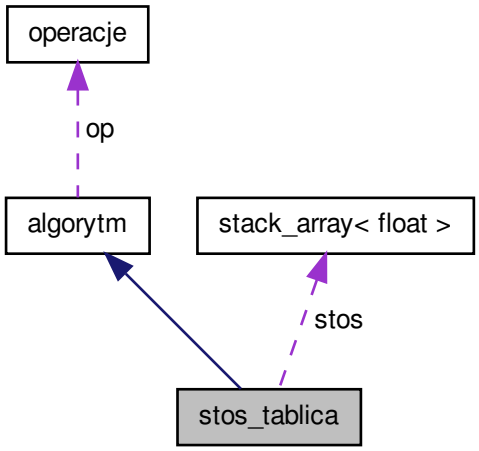


Diagram współpracy dla stos_tablica:



Metody publiczne

- stos_tablica (ifstream &plik1, ifstream &plik2, int N, int M, flag F)

  konstruktor - ustawia flage w zadany stan
- float przelicz ()

  Metoda odpowiada za przetworzenie danych wejsciowych zgodnie z zadanym algorytmem.

Atrybuty prywatne

- stack_array< float > stos

4.11.1 Opis szczegółowy

klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury

Definicja w linii 153 pliku algorytm.hh.

4.11.2 Dokumentacja konstruktora i destruktora

4.11.2.1 stos_tablica::stos_tablica (ifstream & *plik1,* ifstream & *plik2,* int *N,* int *M,* flag *F*) [inline]

konstruktor - ustawia flage w zadany stan

Definicja w linii 159 pliku algorytm.hh.

4.11.3 Dokumentacja funkcji składowych

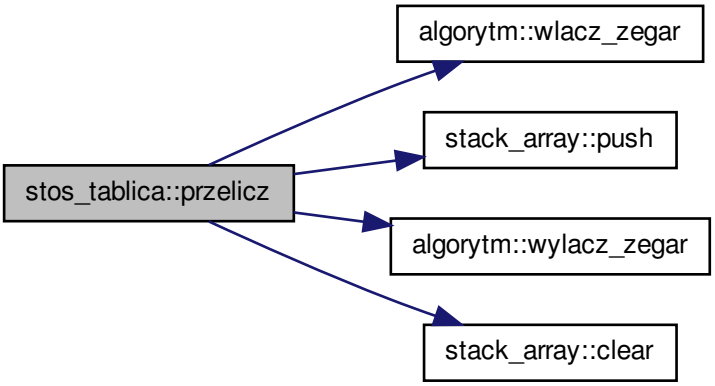
4.11.3.1 float stos_tablica::przelicz ( ) [virtual]

Metoda odpowiada za przetworzenie danych wejsciowych zgodnie z zadanym algorytmem.

Reimplementowana z algorytm.

Definicja w linii 106 pliku algorytm.cpp.

Oto graf wywołań dla tej funkcji:



4.11.4 Dokumentacja atrybutów składowych

4.11.4.1 stack_array<float> stos_tablica::stos [private]

Definicja w linii 154 pliku algorytm.hh.

Dokumentacja dla tej klasy została wygenerowana z plików:

- algorytm.hh
- algorytm.cpp

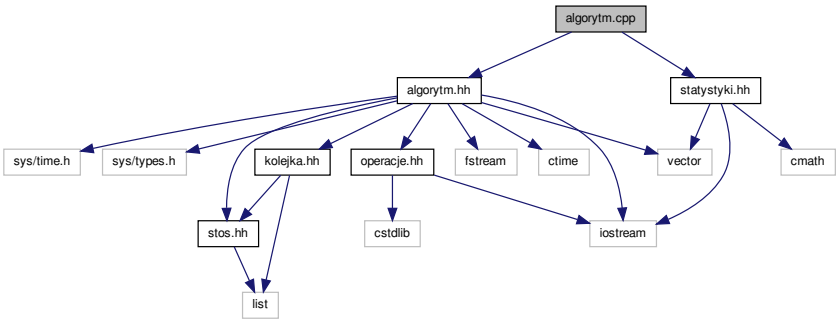
Rozdział 5

Dokumentacja plików

5.1 Dokumentacja pliku algorytm.cpp

plik zawiera definicje metod klas zdefiniowanych w pliku algorytm.hh

`#include "algorytm.hh" #include "statystyki.hh"` Wykres zależności załączania dla algorytm.cpp:



5.1.1 Opis szczegółowy

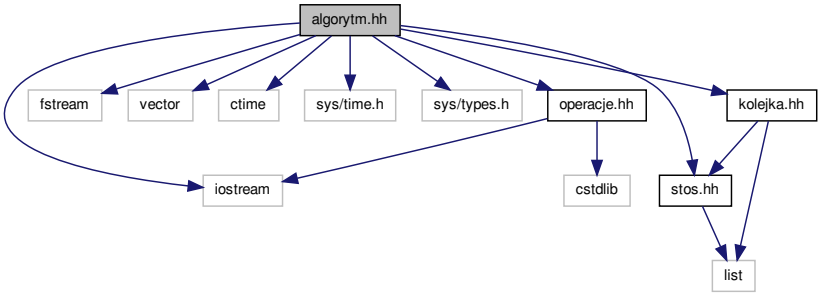
plik zawiera definicje metod klas zdefiniowanych w pliku algorytm.hh

Definicja w pliku algorytm.cpp.

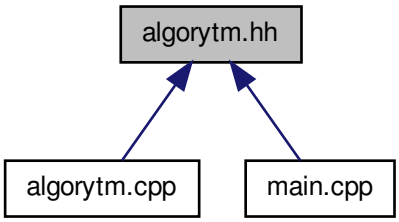
5.2 Dokumentacja pliku algorytm.hh

Definicja klas wykonujacych operacje na zestawie danych wejsciowych.

```
#include <iostream> #include <fstream> #include <vector> x
#include <ctime> #include <sys/time.h> #include <sys/types.-
h>  #include "operacje.hh"  #include "stos.hh"  #include
"kolejka.hh"
```
Wykres zależności załączania dla algorytm.hh:



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



Komponenty

- class algorytm

  Definicja klasy algorytm Jest to klasa bazowa, ktora ma za zadanie wczytac, przetworzyc i porownac dane z plikiem wzorcowym.
- class mnozenie

  modeluje algorytm dokonujacy mnozenia kazdego elementu pliku wejsciowego przez 2
- class stos_tablica

  klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury

- class stos_lista

  klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury

- class kolejka_tablica

  klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury

- class kolejka_lista

  klasa utworzona na potrzeby pomiaru czasu wypełnienia struktury

5.2.1 Opis szczegółowy

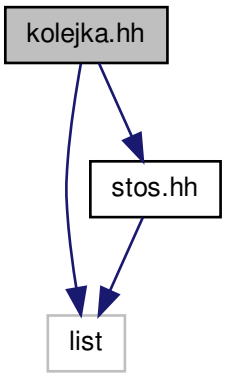
Definicja klas wykonujacych operacje na zestawie danych wejsciowych.

Definicja w pliku algorytm.hh.

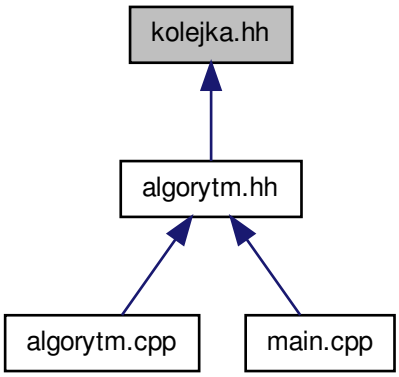
5.3 Dokumentacja pliku kolejka.hh

Plik zawiera definicje klasy Kolejka Zaimplementowanej na 2 sposoby 1. Za pomocą listy. 2. Za pomocą tablicy a. kazdorazowo powiekszajacej swoj rozmiar b. powiekszajacej swoj rozmiar dwukrotnie, gdy kolejka sie przepelni.

`#include <list>` `#include "stos.hh"` Wykres zależności załączania dla kolejka.hh:



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



Komponenty

- class queue_list< TYP >

  Modeluje kolejke oparta na liscie STL.

- class queue_array< TYP >

  Modeluje kolejke w oparciu o tablice.

5.3.1 Opis szczegółowy

Plik zawiera definicje klasy Kolejka Zaimplementowanej na 2 sposoby 1. Za pomocą listy. 2. Za pomocą tablicy a. kazdorazowo powiekszajacej swoj rozmiar b. powiekszajacej swoj rozmiar dwukrotnie, gdy kolejka sie przepelni.

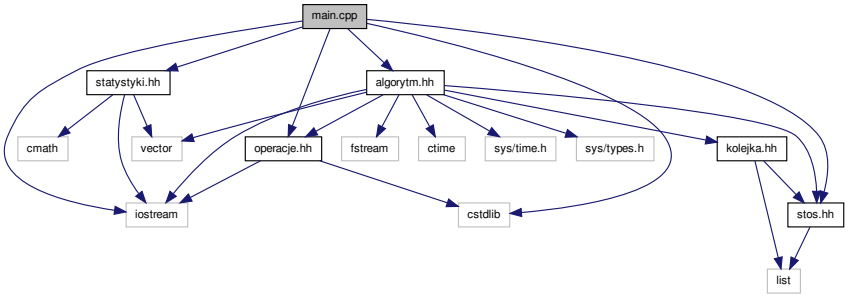
Definicja w pliku kolejka.hh.

5.4 Dokumentacja pliku main.cpp

plik glowny

```
#include <iostream> #include "algorytm.hh" #include "statystyki.-
hh"  #include "operacje.hh"  #include "stos.hh"  #include
```

<cstdlib> Wykres zależności załączania dla main.cpp:



Funkcje

- int main ()

5.4.1 Opis szczegółowy

plik glowny

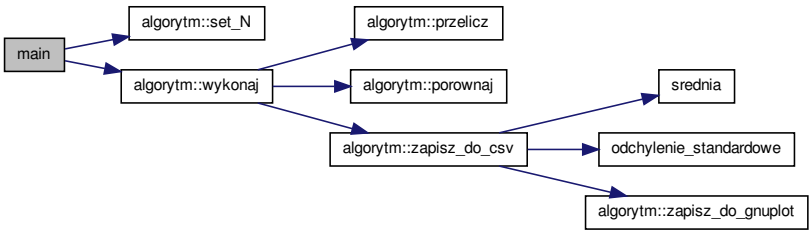
Definicja w pliku main.cpp.

5.4.2 Dokumentacja funkcji

5.4.2.1 int main ( )

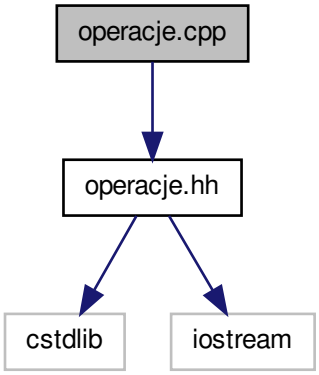
Definicja w linii 13 pliku main.cpp.

Oto graf wywołań dla tej funkcji:



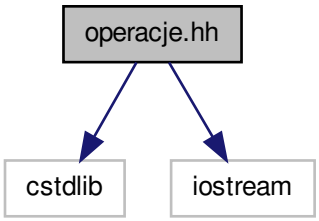
5.5 Dokumentacja pliku operacje.cpp

`#include "operacje.hh"` Wykres zależności załączania dla operacje.cpp:

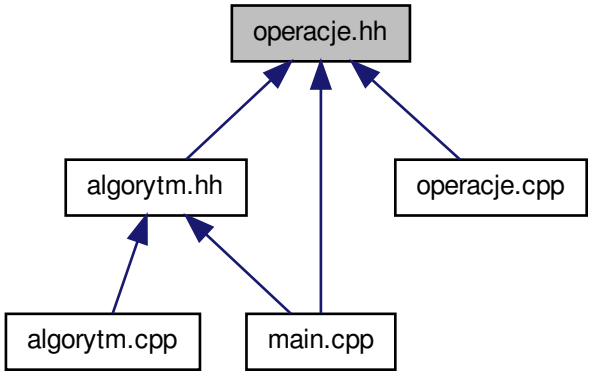


5.6 Dokumentacja pliku operacje.hh

`#include <cstdlib>` `#include <iostream>` Wykres zależności załączania dla operacje.hh:



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



Komponenty

- class operacje

  Klasa modeluje tablice z danymi i metody sluzace do operacji na niej.

Definicje

- #define ROZMIAR 9

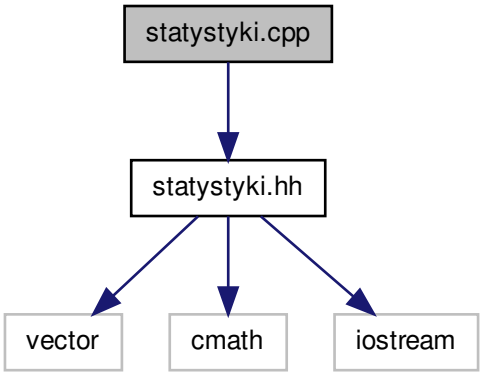
5.6.1 Dokumentacja definicji

5.6.1.1 #define ROZMIAR 9

Definicja w linii 3 pliku operacje.hh.

5.7 Dokumentacja pliku statystyki.cpp

`#include "statystyki.hh"` Wykres zależności załączania dla statystyki.cpp:



Funkcje

- float srednia (float *tab, int rozmiar)

  funckja oblicza wartosc srednia
- float odchylenie_standardowe (float srednia, float *tab, int rozmiar)

  funckja oblicza odchylenie standardowe

5.7.1 Dokumentacja funkcji

5.7.1.1 float odchylenie_standardowe (float *srednia*, float * *tab*, int *rozmiar*)

funckja oblicza odchylenie standardowe

Parametry

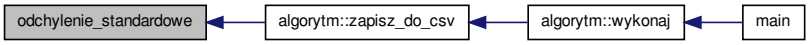
tab	- kontener zawierajacy czasy wykonania algorytmu
srednia	- wartosc srednia
rozmiar	- rozmiar tablicy

Zwraca

odchylenie standardowe

Definicja w linii 16 pliku statystyki.cpp.

Oto graf wywoływań tej funkcji:



5.7.1.2 float srednia (float * *tab*, int *rozmiar*)

funckja oblicza wartosc srednia

Parametry

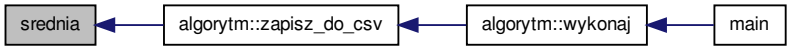
tab	- kontener zawierajacy czasy wykonania algorytmu
rozmiar	- rozmiar tablicy

Zwraca

wartosc srednia

Definicja w linii 3 pliku statystyki.cpp.

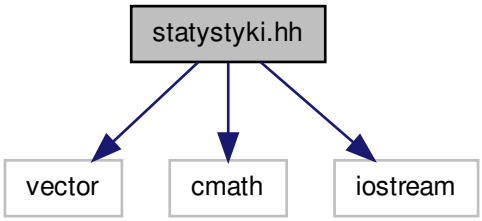
Oto graf wywoływań tej funkcji:



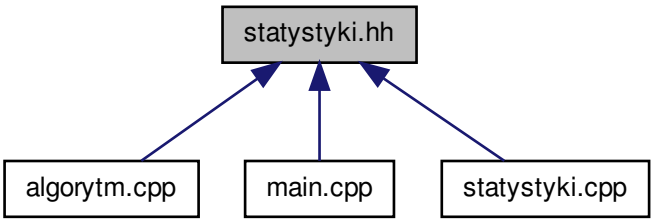
5.8 Dokumentacja pliku statystyki.hh

plik zawiera dekalracje funkcji odpowiedzialnych za przeprowadznaie statystyk

`#include <vector> #include <cmath> #include <iostream>` ×
Wykres zależności załączania dla statystyki.hh:



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



Funkcje

- float srednia (float ∗tab, int rozmiar)

  funckja oblicza wartosc srednia
- float odchylenie_standardowe (float srednia, float ∗tab, int rozmiar)

  funckja oblicza odchylenie standardowe

5.8.1 Opis szczegółowy

plik zawiera dekalracje funkcji odpowiedzialnych za przeprowadznaie statystyk

Definicja w pliku statystyki.hh.

5.8.2 Dokumentacja funkcji

5.8.2.1 float odchylenie_standardowe (float *srednia*, float * *tab*, int *rozmiar*)

funckja oblicza odchylenie standardowe

Parametry

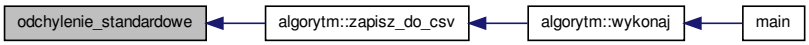
tab	- kontener zawierajacy czasy wykonania algorytmu
srednia	- wartosc srednia
rozmiar	- rozmiar tablicy

Zwraca

odchylenie standardowe

Definicja w linii 16 pliku statystyki.cpp.

Oto graf wywoływań tej funkcji:



5.8.2.2 float srednia (float * *tab*, int *rozmiar*)

funckja oblicza wartosc srednia

Parametry

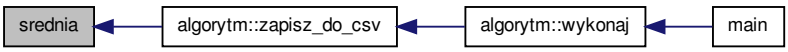
tab	- kontener zawierajacy czasy wykonania algorytmu
rozmiar	- rozmiar tablicy

Zwraca

wartosc srednia

Definicja w linii 3 pliku statystyki.cpp.

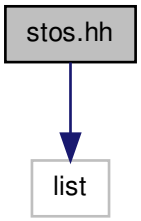
Oto graf wywoływań tej funkcji:



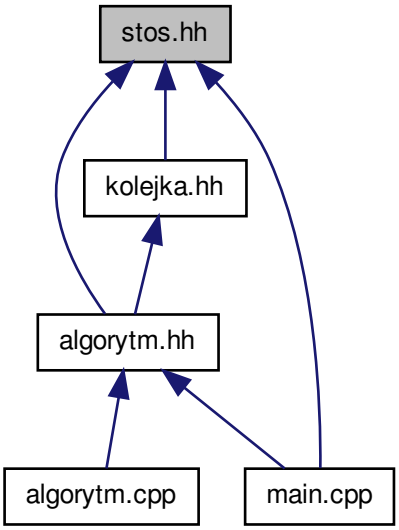
5.9 Dokumentacja pliku stos.hh

Plik zawiera definicje klasy `Stos` Zaimplementowana na 2 sposoby 1. Za pomocą listy. 2. Za pomocą tablicy a. kazdorazowo powiekszajacej swoj rozmiar b. powiekszajacej swoj rozmiar dwukrotnie, gdy stos sie przepelni.

`#include <list>` Wykres zależności załączania dla stos.hh:



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



Komponenty

- class stack_list< TYP >

  Modeluje stos oparty na liscie STL.
- class stack_array< TYP >

  Modeluje stos w oparciu o tablice.

Wyliczenia

- enum flag { plus1, x2 }

  typ wyliczeniowy sluzacy do ustawienia sposobu zwiekszania pamieci

5.9.1 Opis szczegółowy

Plik zawiera definicje klasy `Stos` Zaimplementowana na 2 sposoby 1. Za pomocą listy. 2. Za pomocą tablicy a. kazdorazowo powiekszajacej swoj rozmiar b. powiekszajacej swoj rozmiar dwukrotnie, gdy stos sie przepelni.

Definicja w pliku stos.hh.

5.9.2 Dokumentacja typów wyliczanych

5.9.2.1 enum flag

typ wyliczeniowy sluzacy do ustawienia sposobu zwiekszania pamieci

Wartości wyliczeń:

plus1

x2

Definicja w linii 17 pliku stos.hh.

5.10 Dokumentacja pliku strona.dox