Tablica asocjacyjna

Generated by Doxygen 1.8.1.2

Mon Apr 7 2014 00:10:35

Spis treści

Rozdział 1

Class Index

1.1 Class List

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

Rozdział 2

Class Documentation

2.1 Dzialanie Class Reference

Klasa modelujaca gowna czesc programu.

`#include <dzialanie.hh>`

Public Member Functions

- void wczytajDaneWejsciowe (string nazwa)

  metoda wczytuje dane do tablicy znajdujacej sie w zmiennej wejscie.
- LARGE_INTEGER wlaczStoper ()

  Metoda uruchamia pomiar czasu.
- LARGE_INTEGER wylaczStoper ()

  Metoda konczy pomiar czasu.
- void wykonajAlgorytm ()

  Metoda wykonuje algorytm na danych wejsciowych (tablicy)
- bool sprawdz ()

  Metoda sprawdza poprawnosc algorytmu.
- int uruchom (string nazwa)

  Metoda wykonuje jednorazowy test algorytmu Metoda: -wlacza zegar -wykonuje algorytm -wylacza zegar -sprawdza poprawnosc algorytmu return czas wykonywania algorytmu w milisekundach.
- void Quicksort (Kontener ∗tab, int lewy, int prawy)

  Metoda implementujca sortowanie szybkie.
- void Quicksort_lepiej (Kontener ∗tab, int lewy, int prawy)

  Metoda implementujca usprawnione sortowanie szybkie Wybor piwotu jest losowy, co zmniejsza szanse na pojawienie sie przypadku pesymistycznego.
- Kontener heap (Kontener tab)

  Metoda implementujca sortowanie przez kopcowanie.
- void uporzadkuj_kopiec (Kontener ∗kopiec)

  Metoda przywraca sturkturpca jesli ta zostala zaburzona.
- Kontener merge (Kontener lewy, Kontener prawy)

  Metoda implementujca scalanie dwoch tablic Scalanie czyli laczenie dwoch posortowanych tablic w jedna(rowniez posortowana)
- Kontener MergeSort (Kontener tab)

  Metoda implementujca sortowanie przez scalanie.
- int **wez_rozmiar** ()

2.1.1 Detailed Description

Klasa modelujaca gowna czesc programu.

Klasa modeluje glowna czesc porgramu, ktorego zadaniem jest: -wczytanie danych -zmierzenie czasu dzialania algorytmu -sprawdzenie poprawnosci tego algorytmu, majac oczekiwany wynik

2.1.2 Member Function Documentation

2.1.2.1 Kontener Dzialanie::heap (Kontener *tab*)

Metoda implementujca sortowanie przez kopcowanie.

Parameters

tab	- tablica do posortowania

Returns

metoda zwraca posortowana tablice

2.1.2.2 Kontener Dzialanie::merge (Kontener *lewy,* Kontener *prawy*)

Metoda implementujca scalanie dwoch tablic Scalanie czyli laczenie dwoch posortowanych tablic w jedna(rowniez posortowana)

Parameters

lewy	- element pierwszy
prawy	- element drugi

Returns

metoda zwraca scalony zbior

2.1.2.3 Kontener Dzialanie::MergeSort (Kontener *tab*)

Metoda implementujca sortowanie przez scalanie.

Parameters

tab	- tablica do posortowania

Returns

metoda zwraca posortowana tablice

2.1.2.4 void Dzialanie::Quicksort (Kontener * *tab,* int *lewy,* int *prawy*)

Metoda implementujca sortowanie szybkie.

Parameters

tab	- wskaznik na tablice do posortowania
lewy	- indeks poczatku tablicy
prawy	- indeks konca tablicy

2.1.2.5 void Dzialanie::Quicksort_lepiej (Kontener * *tab,* int *lewy,* int *prawy*)

Metoda implementujca usprawnione sortowanie szybkie Wybor piwotu jest losowy, co zmniejsza szanse na pojawienie sie przypadku pesymistycznego.

Parameters

tab	- wskaznik na tablice do posortowania
lewy	- indeks poczatku tablicy
prawy	- indeks konca tablicy

2.1.2.6 bool Dzialanie::sprawdz ( )

Metoda sprawdza poprawnosc algorytmu.

Wczytywane sa poprawne dane wynikowe, a nastepnie sa one porownywane z tymi otrzymanymi przez wykonanie algorytmu

Returns

0 - gdy algorytm jest poprawny, -1 - gdy nie.

2.1.2.7 void Dzialanie::uporzadkuj_kopiec (Kontener * *kopiec*)

Metoda przywraca sturkturpca jesli ta zostala zaburzona.

Parameters

kopiec	- wskaznik na kopiec do uporzadkowania

2.1.2.8 void Dzialanie::wczytajDaneWejsciowe (string *nazwa*)

metoda wczytuje dane do tablicy znajdujacej sie w zmiennej wejscie.

Format danych w pliku jest nastepujacy: pierwszy wiersz - ilosc elementow, a nastepnie w kolumnie kolejne wartosci tablicy.

Parameters

nazwaPliku	- nazwa pliku do otwarcia

Returns

void

2.1.2.9 LARGE_INTEGER Dzialanie::wlaczStoper ( )

Metoda uruchamia pomiar czasu.

Czas jest mierzony w milisekundach

Returns

czas, w ktorym zostal wlaczony stoper

2.1.2.10 void Dzialanie::wykonajAlgorytm ( )

Metoda wykonuje algorytm na danych wejsciowych (tablicy)

Algorytm do wykonania : pomnoz kazdy element razy 2.

Returns

void

2.1.2.11 LARGE_INTEGER Dzialanie::wylaczStoper ( )

Metoda konczy pomiar czasu.

Czas jest mierzony w milisekundach

Returns

czas, w ktorym stoper zostal wylaczony

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

- inc/dzialanie.hh
- src/dzialanie.cpp

2.2 kolejka_list Class Reference

Klasa modelujaca strukture koleki Kolejka jest zbudowana w oparciu o liste.

`#include <kolejka_list.hh>`

Public Member Functions

- void enqueue (int element)

  Metoda dodajaca element na stos Metoda dodaje element na koniec listy.
- void dequeue (int *a)

  Metoda usuwajaca element ze stos Metoda usuwa element znajdujacy sie na koncu listy. Gdy stos jest pusty, wyswietlony zostaje blad mowiacy ze brak danych do sciagniecia.
- bool isempty ()

  Metoda sprawdzajaca czy stos jest pusty. Metoda sprawdza czy stos jest pusty.
- int size ()

  Metoda zwracajaca rozmiar stosu Metoda zwraca rozmiar listy.
- void **wyswietl** ()

2.2.1 Detailed Description

Klasa modelujaca strukture koleki Kolejka jest zbudowana w oparciu o liste.

2.2.2 Member Function Documentation

2.2.2.1 void kolejka_list::dequeue (int * *a*)

Metoda usuwajaca element ze stos Metoda usuwa element znajdujacy sie na koncu listy. Gdy stos jest pusty, wyswietlony zostaje blad mowiacy ze brak danych do sciagniecia.

Parameters

a	- wskaznik do zmiennej, do ktorej sciagamy wartosc.

2.2.2.2 void kolejka_list::enqueue (int *element*)

Metoda dodajaca element na stos Metoda dodaje element na koniec listy.

Parameters

element	- element, ktory zostanie dodany do stosu

2.2.2.3 bool kolejka_list::isempty ( )

Metoda sprawdzajaca czy stos jest pusty. Metoda sprawdza czy stos jest pusty.

Returns

lista.empty();

2.2.2.4 int kolejka_list::size ( )

Metoda zwracajaca rozmiar stosu Metoda zwraca rozmiar listy.

Returns

rozmiar stosu.

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

- inc/kolejka_list.hh
- src/kolejka_list.cpp

2.3 kolejka_tab Class Reference

Klasa modelujaca strukture koleki Kolejka jest zbudowana w oparciu o liste.

`#include <kolejka_tab.hh>`

Public Member Functions

- **kolejka_tab** (int zwieksz)
- void enqueue (int element)

  Metoda dodajaca element na stos Metoda dodaje element na koniec listy.
- void dequeue (int *a)

  Metoda usuwajaca element ze stos Metoda usuwa element znajdujacy sie na koncu listy. Gdy stos jest pusty, wyswietlony zostaje blad mowiacy ze brak danych do sciagniecia.
- bool isempty ()

  Metoda sprawdzajaca czy stos jest pusty. Metoda sprawdza czy stos jest pusty.
- int size ()

  Metoda zwracajaca rozmiar stosu Metoda zwraca rozmiar listy.
- void **wyswietl** ()

2.3.1 Detailed Description

Klasa modelujaca strukture koleki Kolejka jest zbudowana w oparciu o liste.

2.3.2 Member Function Documentation

2.3.2.1 void kolejka_tab::dequeue (int * *a*)

Metoda usuwajaca element ze stos Metoda usuwa element znajdujacy sie na koncu listy. Gdy stos jest pusty, wyswietlony zostaje blad mowiacy ze brak danych do sciagniecia.

Parameters

a	- wskaznik do zmiennej, do ktorej sciagamy wartosc.

2.3.2.2 void kolejka_tab::enqueue (int *element*)

Metoda dodajaca element na stos Metoda dodaje element na koniec listy.

Parameters

element	- element, ktory zostanie dodany do stosu

2.3.2.3 bool kolejka_tab::isempty ( )

Metoda sprawdzajaca czy stos jest pusty. Metoda sprawdza czy stos jest pusty.

Returns

lista.empty();

2.3.2.4 int kolejka_tab::size ( )

Metoda zwracajaca rozmiar stosu Metoda zwraca rozmiar listy.

Returns

rozmiar stosu.

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

- inc/kolejka_tab.hh
- src/kolejka_tab.cpp

2.4 Kontener Class Reference

Klasa Dane.

`#include <kontener.hh>`

Public Member Functions

- unsigned int wez_rozmiar ()

  Metoda zwracajaca rozmiar tablicy.
- vector< int > & wez_dane ()

  Metoda zwracajaca referencje do tablicy danych Metoda pozwala na dostep do tablicy i jej modyfikacje.
- void wczytajDane (string nazwaPliku)

  metoda wczytuje dane do tablicy z pliku
- void zamien_elementy (unsigned int i, unsigned int j)

  Metoda zamienia ze soba dwa elementy tablicy.
- void odwroc_kolejnosc ()

  Metoda odwraca zawartosc tablicy.
- void dodaj_element (int e)

  Metoda dodaje element na koniec tablicy.
- void **usun_z_konca** ()
- void **usun_z_poczatku** ()
- void dodaj_elementy (Kontener tab)

  Metoda dodaje na koniec tablicy zawartosc innej tablicy.
- int & operator[] (int index)

  Operator indeksujacy tablice.
- const int & **operator[]** (int el) const
- Kontener & operator+ (Kontener tab)

  Operator dodawania tablic Operator pozwala na dodanie 2 tablic.
- Kontener & operator= (Kontener tab)

  Operator przypisania Operator pozwala na przypisanie do tablicy zawartosci innej tablicy.
- bool operator== (Kontener tab)

  Operator porownania 2 tablic Operator pozwala na porownanie 2 tablic. Sprawdza on czy sa takie same pod wzgledem zawartosci.

Friends

- ostream & operator<< (ostream &out, Kontener Tab)

  Operator wwypisywania Metoda pozwala na wypisanie zawartosci tablicy na standardowe wyjscie.

2.4.1 Detailed Description

Klasa Dane.

Klasa posiada 2 pola: -tablice (vector), -rozmiar tablicy.

2.4.2 Member Function Documentation

2.4.2.1 void Kontener::dodaj_element (int *e*)

Metoda dodaje element na koniec tablicy.

Parameters

e	- wartosc elementu

2.4.2.2 void Kontener::dodaj_elementy (Kontener *tab*)

Metoda dodaje na koniec tablicy zawartosc innej tablicy.

Parameters

tablica,ktora	bedzie dodana na koniec

2.4.2.3 Kontener & Kontener::operator+ (Kontener *tab*)

Operator dodawania tablic Operator pozwala na dodanie 2 tablic.

```
\param tablica do dodania
```

Returns

dwie polaczone tablice

2.4.2.4 Kontener & Kontener::operator= (Kontener *tab*)

Operator przypisania Operator pozwala na przypisanie do tablicy zawartosci innej tablicy.

Parameters

tablica,ktora	przypisujemy

2.4.2.5 bool Kontener::operator== (Kontener *tab*)

Operator porownania 2 tablic Operator pozwala na porownanie 2 tablic. Sprawdza on czy sa takie same pod wzgledem zawartosci.

Parameters

tablica,z	ktora bedziemy porownywac

Returns

true - tablice sa identyczne
false - tablice sa rozne

2.4.2.6 int & Kontener::operator[] (int *index*)

Operator indeksujacy tablice.

Parameters

index	- indeks, ktorego referencja zostanie zwrocona

Returns

referencja do zadanego indeksu

2.4.2.7 void Kontener::wczytajDane (string *nazwaPliku*)

metoda wczytuje dane do tablicy z pliku

Format danych w pliku jest nastepujacy: pierwszy wiersz - ilosc elementow, a nastepnie w kolumnie kolejne wartosci tablicy.

Parameters

nazwaPliku	- nazwa pliku do otwarcia

Returns

void

2.4.2.8 vector<int>& Kontener::wez_dane ( ) [inline]

Metoda zwracajaca referencje do tablicy danych Metoda pozwala na dostep do tablicy i jej modyfikacje.

Returns

referencja do tablicy

2.4.2.9 unsigned int Kontener::wez_rozmiar ( ) [inline]

Metoda zwracajaca rozmiar tablicy.

Returns

rozmiar tablicy

2.4.2.10 void Kontener::zamien_elementy (unsigned int *i*, unsigned int *j*)

Metoda zamienia ze soba dwa elementy tablicy.

Parameters

i,j	- indeksy, ktore zostana zamienione

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

- inc/kontener.hh
- src/kontener.cpp

2.5 Para< Wartosc > Class Template Reference

Klasa modeluje pare klucz-wartosc, ktora jest podstawowym elementem tablicy asocjacyjnej Zar typ Klucz jak i Wartosc musz miec konstruktory parametryczne, ktore inicjuja ich wartosci.

`#include <slownik.hh>`

Public Member Functions

- Para ()

Konstruktor inicjujacy wartosci key i val.

- Para (const std::string &key)

  Konstruktor inicjujacy wartosci key i val.

- Para (const std::string &key, const Wartosc &val)

  Konstruktor inicjujacy wartosci key i val.

- std::string & klucz ()

  Metoda zwraca referencje do pola key, dzieki niej mozemy modyfikowac jego wartosc.

- Wartosc & wartosc ()

  Metoda zwraca referencje do pola val, dzieki niej mozemy modyfikowac jego wartosc.

- const std::string & klucz () const

  Metoda zwraca wartosc pola key, sluzy ona do odczytu tej wartosci.

- const Wartosc & wartosc () const

  Metoda zwraca wartosc pola val, sluzy ona do odczytu tej wartosci.

2.5.1 Detailed Description

template<typename Wartosc>class Para< Wartosc >

Klasa modeluje pare klucz-wartosc, ktora jest podstawowym elementem tablicy asocjacyjnej Zar typ Klucz jak i Wartosc musz miec konstruktory parametryczne, ktore inicjuja ich wartosci.

2.5.2 Member Function Documentation

2.5.2.1 template<typename Wartosc> std::string& Para< Wartosc >::klucz ( ) [inline]

Metoda zwraca referencje do pola key, dzieki niej mozemy modyfikowac jego wartosc.

Returns

referencja do pola key

2.5.2.2 template<typename Wartosc> const std::string& Para< Wartosc >::klucz ( ) const [inline]

Metoda zwraca wartosc pola key, sluzy ona do odczytu tej wartosci.

Returns

wartosc pola key

2.5.2.3 template<typename Wartosc> Wartosc& Para< Wartosc >::wartosc ( ) [inline]

Metoda zwraca referencje do pola val, dzieki niej mozemy modyfikowac jego wartosc.

Returns

referencja do pola val

2.5.2.4 template<typename Wartosc> const Wartosc& Para< Wartosc >::wartosc ( ) const [inline]

Metoda zwraca wartosc pola val, sluzy ona do odczytu tej wartosci.

Returns

wartosc pola val

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

- inc/slownik.hh

2.6 stos_list Class Reference

Klasa modelujaca strukture stosu Stos jest zbudowany w oparciu o liste.

`#include <stos_list.hh>`

Public Member Functions

- void push (int element)

  Metoda dodajaca element na stos Metoda dodaje element na koniec listy.
- void pop (int *a)

  Metoda usuwajaca element ze stos Metoda usuwa element znajdujacy sie na koncu listy. Gdy stos jest pusty, wyswietlony zostaje blad mowiacy ze brak danych do sciagniecia.
- bool isempty ()

  Metoda sprawdzajaca czy stos jest pusty. Metoda sprawdza czy stos jest pusty.
- int size ()

  Metoda zwracajaca rozmiar stosu Metoda zwraca rozmiar listy.
- void **wyswietl** ()

2.6.1 Detailed Description

Klasa modelujaca strukture stosu Stos jest zbudowany w oparciu o liste.

2.6.2 Member Function Documentation

2.6.2.1 bool stos_list::isempty ( )

Metoda sprawdzajaca czy stos jest pusty. Metoda sprawdza czy stos jest pusty.

Returns

lista.empty();

2.6.2.2 void stos_list::pop (int * *a*)

Metoda usuwajaca element ze stos Metoda usuwa element znajdujacy sie na koncu listy. Gdy stos jest pusty, wyswietlony zostaje blad mowiacy ze brak danych do sciagniecia.

Parameters

a	- wskaznik do zmiennej, do ktorej sciagamy wartosc.

2.6.2.3 void stos_list::push (int *element*)

Metoda dodajaca element na stos Metoda dodaje element na koniec listy.

Parameters

element	- element, ktory zostanie dodany do stosu

2.6.2.4 int stos_list::size ( )

Metoda zwracajaca rozmiar stosu Metoda zwraca rozmiar listy.

Returns

rozmiar stosu.

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

- inc/stos_list.hh
- src/stos_list.cpp

2.7 stos_tab Class Reference

Klasa modelujaca strukture stosu Stos jest zbudowany w oparciu o tablice.

`#include <stos_tab.hh>`

Public Member Functions

- **stos_tab** (int zwieksz)
- void push (int element)

  Metoda dodajaca element na stos Metoda dodaje element na koniec tablicy. W przypadku gdy tablica jest pelna zostaje ona powiekszona o 1 element.
- void pop (int *a)

  Metoda usuwajaca element ze stos Metoda usuwa element znajdujacy sie na koncu tablicy. Z kazdorazowym pobraniem danych rozmiar tablicy jest zmniejszany o 1.
- bool isempty ()

  Metoda sprawdzajaca czy stos jest pusty. Metoda sprawdza czy stos jest pusty. Sprawdzenie polega na odczytaniu zmiennej rozmiar,
  ktora pamieta ile jest elementow na stosie.
- int size ()

  Metoda zwracajaca rozmiar stosu Metoda zwraca rozmiar stosu. Rozmiar stosu przechowywany jest w zmiennej rozmiar.
- void **wyswietl** ()

2.7.1 Detailed Description

Klasa modelujaca strukture stosu Stos jest zbudowany w oparciu o tablice.

2.7.2 Member Function Documentation

2.7.2.1 bool stos_tab::isempty ( )

Metoda sprawdzajaca czy stos jest pusty. Metoda sprawdza czy stos jest pusty. Sprawdzenie polega na odczytaniu zmiennej rozmiar,

ktora pamieta ile jest elementow na stosie.

Returns

true - rozmiar == 0
false - rozmiar > 0.

2.7.2.2 void stos_tab::pop (int * *a*)

Metoda usuwajaca element ze stos Metoda usuwa element znajdujacy sie na koncu tablicy. Z kazdorazowym pobraniem danych rozmiar tablicy jest zmniejszany o 1.

Parameters

a	- wskaznik do zmiennej, do ktorej sciagamy wartosc.

2.7.2.3 void stos_tab::push (int *element*)

Metoda dodajaca element na stos Metoda dodaje element na koniec tablicy. W przypadku gdy tablica jest pelna zostaje ona powiekszona o 1 element.

Parameters

element	- element, ktory zostanie dodany do stosu

2.7.2.4 int stos_tab::size ( )

Metoda zwracajaca rozmiar stosu Metoda zwraca rozmiar stosu. Rozmiar stosu przechowywany jest w zmiennej rozmiar.

Returns

rozmiar stosu.

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

- inc/stos_tab.hh
- src/stos_tab.cpp

2.8 Tablica_asocjacyjna< Wartosc > Class Template Reference

Klasa modeluje pojecie tablicy asocjacyjnej Tablica asocjacyjna jest to tablica, w ktorej do wartosci mozemy sie odwolywac poprzez klucze. Odwolanie poprez indeks rowniez jest mozliwe.

```
#include <slownik.hh>
```

Public Member Functions

- Tablica_asocjacyjna ()

  Meteoda dodaje pare klucz-warotsc do tablicy.
- void dodaj (std::string klucz, Wartosc wartosc)

  Metoda dodaje pare klucz-wartosc do tablicy W przypadku gdy dany klucz juz istnieje to wartosc, ktora jest z nim skojarzona zostaje zastapiona ta
  podana w argumencie "wartosc" Po kazdorazowym dodaniu nowego klucza, tablica jest sortrowana alfabetycznie wzgledem nazw kluczy. Takie rozwiazanie daje mozliwosc przeszukiwania binarnego co powoduje ze dostep do elementu jest w czasie logn.
- void zmien (string klucz, Wartosc wartosc)

  Metoda pobiera wartosc przypisana pod zadanym kluczem.
- Wartosc **pobierz** (std::string klucz) const
- void usun (string klucz)

  Metoda usuwa pozycje zawierajaca podany klucz. Gdy klucz istnieje, to jest usuwany, a flaga czy_istnieje ustawiana na wartosc true, przeciwnie flaga ma wartosc false.
- string znajdz (Wartosc wartosc)

  Metoda znajduje pozycje(nazwe klucza), pod ktora znajduje sie podana wartosc.
- const bool czy_znalazlo () const

  Metoda zwraca stan flagi czy_istnieje. Flaga czy istnieje jest ustawiana za kazdym razem gdy wykonujemy funkcje, ktora przeszukuje tablice po kluczach. Gdy zadany klucz istnieje to flaga jest ustawiana. Natomiast gdy danego klucza nie ma w zbiorze to flaga jest zerowana.
- const bool **czy_pusta** ()
- const int **rozmiar** ()
- Wartosc **operator[]** (string klucz) const
- Wartosc & **operator[]** (string klucz)

2.8.1 Detailed Description

template<typename Wartosc>class Tablica_asocjacyjna< Wartosc >

Klasa modeluje pojecie tablicy asocjacyjnej Tablica asocjacyjna jest to tablica, w ktorej do wartosci mozemy sie odwolywac poprzez klucze. Odwolanie poprez indeks rowniez jest mozliwe.

2.8.2 Constructor & Destructor Documentation

2.8.2.1 template<typename Wartosc > Tablica_asocjacyjna< Wartosc >::Tablica_asocjacyjna ( ) [inline]

Meteoda dodaje pare klucz-warotsc do tablicy.

zmienna pomocnicza, ktora sluzy do okreslania czy wartosc o podanym kluczu

znajduje sie w zbiorze

Parameters

klucz	
wartosc	

2.8.3 Member Function Documentation

2.8.3.1 template<typename Wartosc > const bool Tablica_asocjacyjna< Wartosc >::czy_znalazlo ( ) const [inline]

Metoda zwraca stan flagi czy_istnieje. Flaga czy istnieje jest ustawiana za kazdym razem gdy wykonujemy funkcje, ktora przeszukuje tablice po kluczach. Gdy zadany klucz istnieje to flaga jest ustawiana. Natomiast gdy danego klucza nie ma w zbiorze to flaga jest zerowana.

Returns

true - zadany klucz istnieje w zbiorze
false - klucz nie istnieje.

2.8.3.2 template<typename Wartosc > void Tablica_asocjacyjna< Wartosc >::dodaj (std::string *klucz,* Wartosc *wartosc*)

Metoda dodaje pare klucz-wartosc do tablicy W przypadku gdy dany klucz juz istnieje to wartosc, ktora jest z nim skojarzona zostaje zastapiona ta

podana w argumencie "wartosc" Po kazdorazowym dodaniu nowego klucza, tablica jest sortrowana alfabetycznie wzgledem nazw kluczy. Takie rozwiazanie daje mozliwosc przeszukiwania binarnego co powoduje ze dostep do elementu jest w czasie logn.

Parameters

in	*klucz-nazwa*	klucza
in	*wartosc-wartosc*	do zapisania

2.8.3.3 template<typename Wartosc > void Tablica_asocjacyjna< Wartosc >::usun (string *klucz*)

Metoda usuwa pozycje zawierajaca podany klucz. Gdy klucz istnieje, to jest usuwany, a flaga czy_istnieje ustawiana na wartosc true, przeciwnie flaga ma wartosc false.

Parameters

in	*klucz-klucz*	do usuniecia

2.8.3.4 template<typename Wartosc > void Tablica_asocjacyjna< Wartosc >::zmien (string *klucz,* Wartosc *wartosc*) [inline]

Metoda pobiera wartosc przypisana pod zadanym kluczem.

Jesli klucz nie istnieje to flaga czy_istnieje jest ustawiana na false, w przeciwnym wypadku ma ona wartosc true. Po wywolaniu metody "pobierz" mozna sprawdzic stan flagi metoda czy_znalazlo() i jego podstawie okreslic

czy operacja zostala wykonana poprawnie.

Parameters

in	*klucz-*	nazwa klucza, ktorego szukamy

Returns

wartosc spod zadanego klucza

2.8.3.5 template<typename Wartosc > string Tablica_asocjacyjna< Wartosc >::znajdz (Wartosc *wartosc*)

Metoda znajduje pozycje(nazwe klucza), pod ktora znajduje sie podana wartosc.

Returns

gdy wartosc istnieje: nazwa klucza,
gdy nie istnieje: pusta zmienna string

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

- inc/slownik.hh

2.9 Tester Class Reference

Klasa Tester Klasa modeluje narzedzie do benchmarkowania algorytmu. Poprzez wielokrotne wynonywanie algorytmu mozemy wyciagnac sredni czas dzialania, co jest dokladniejszym pomiarem.

`#include <tester.hh>`

Public Member Functions

- Tester ()

  konstruktor klasy Konstruktor inicjuje wartosci: -powtorzenia (ile razy wykonywac algorytm) -ilosc (ile mamy zestawow dancyh) -wejscie (nazwa pliku z danymi wejsciowymi) -wynik (nazwa pliku z poprawnym wynikiem algorytmu)
- void otworzPlik (string nazwa)

  Metoda otwierajaca referencje do pliku CSV.
- void zamknijPlik ()

  Metoda otwierajaca referencje do pliku CSV.
- void symulacja ()

  Metoda symulujaca badanie algorytmu Metoda wykonuje symulacje dzialania algorytmu. Wykonuje algorytm zadana liczbe razy dla zadanych zestawow danych. Wyniki zapisuje do pliku CSV o nazwie rezultat.csv Format zapisu: rozmiar_problemu,ilosc_powtorzen,sredni_czas.
- void zamienNazwy (int numer)

  Metoda aktualizuje nazwe plikow wejsciowych Format plikow wejsciowych jest scisle okreslony. Dane wejsciowe: wejscie%.txt, gdzie % to numer zestawu Dane wynikowe: wynik%.txt, gdzie % to numer zestawu Zestawy nalazy numerowac od 0.

2.9.1 Detailed Description

Klasa Tester Klasa modeluje narzedzie do benchmarkowania algorytmu. Poprzez wielokrotne wynonywanie algorytmu mozemy wyciagnac sredni czas dzialania, co jest dokladniejszym pomiarem.

2.9.2 Member Function Documentation

2.9.2.1 void Tester::zamienNazwy (int *numer*)

Metoda aktualizuje nazwe plikow wejsciowych Format plikow wejsciowych jest scisle okreslony. Dane wejsciowe: wejscie%.txt, gdzie % to numer zestawu Dane wynikowe: wynik%.txt, gdzie % to numer zestawu Zestawy nalazy numerowac od 0.

Parameters

numer	- % zostaje zamieniony na wartosc 'numer'

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

- inc/tester.hh
- src/tester.cpp

Skorowidz