

Algorytmy sortowania

Generated by Doxygen 1.8.1.2

Wed Apr 2 2014 20:29:05

Contents

1	Class Index	1
1.1	Class List	1
2	Class Documentation	3
2.1	Dzialanie Class Reference	3
2.1.1	Detailed Description	4
2.1.2	Member Function Documentation	4
2.1.2.1	heap	4
2.1.2.2	merge	4
2.1.2.3	MergeSort	4
2.1.2.4	Quicksort	4
2.1.2.5	sprawdz	5
2.1.2.6	uporzadkuj_kopiec	5
2.1.2.7	wczytajDaneWejsciowe	5
2.1.2.8	wlaczStoper	5
2.1.2.9	wykonajAlgorytm	5
2.1.2.10	wylaczStoper	6
2.2	kolejka_list Class Reference	6
2.2.1	Detailed Description	6
2.2.2	Member Function Documentation	6
2.2.2.1	dequeue	6
2.2.2.2	enqueue	6
2.2.2.3	isempty	7
2.2.2.4	size	7
2.3	kolejka_tab Class Reference	7
2.3.1	Detailed Description	7
2.3.2	Member Function Documentation	8
2.3.2.1	dequeue	8
2.3.2.2	enqueue	8
2.3.2.3	isempty	8
2.3.2.4	size	8

2.4	Kontener Class Reference	8
2.4.1	Detailed Description	9
2.4.2	Member Function Documentation	9
2.4.2.1	dodaj_element	9
2.4.2.2	dodaj_elementy	9
2.4.2.3	operator+	10
2.4.2.4	operator=	10
2.4.2.5	operator==	10
2.4.2.6	operator[]	10
2.4.2.7	wczytajDane	10
2.4.2.8	wez_dane	11
2.4.2.9	wez_rozmiar	11
2.4.2.10	zamien_elementy	11
2.5	stos_list Class Reference	11
2.5.1	Detailed Description	12
2.5.2	Member Function Documentation	12
2.5.2.1	isempty	12
2.5.2.2	pop	12
2.5.2.3	push	12
2.5.2.4	size	12
2.6	stos_tab Class Reference	12
2.6.1	Detailed Description	13
2.6.2	Member Function Documentation	13
2.6.2.1	isempty	13
2.6.2.2	pop	13
2.6.2.3	push	13
2.6.2.4	size	14
2.7	Tester Class Reference	14
2.7.1	Detailed Description	14
2.7.2	Member Function Documentation	14
2.7.2.1	zamienNazwy	14

Chapter 1

Class Index

1.1 Class List

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

Dzialanie	Klasa modelujaca gowna czesc programu	3
kolejka_list	Klasa modelujaca strukture koleki Kolejka jest zbudowana w oparciu o liste	6
kolejka_tab	Klasa modelujaca strukture koleki Kolejka jest zbudowana w oparciu o liste	7
Kontener	Klasa Dane	8
stos_list	Klasa modelujaca strukture stosu Stos jest zbudowany w oparciu o liste	11
stos_tab	Klasa modelujaca strukture stosu Stos jest zbudowany w oparciu o tablice	12
Tester	Klasa Tester Klasa modeluje narzedzie do benchmarkowania algorytmu. Poprzez wielokrotne wynonywanie algorytmu mozemy wyciagnac sredni czas dzialania, co jest dokladniejszym pomiarem	14

Chapter 2

Class Documentation

2.1 Działanie Class Reference

Klasa modelująca gówna część programu.

```
#include <dzialanie.hh>
```

Public Member Functions

- void [wczytajDaneWejscowe](#) (string nazwa)
metoda wczytuje dane do tablicy znajdujacej sie w zmiennej wejscie.
- LARGE_INTEGER [włączStoper](#) ()
Metoda uruchamia pomiar czasu.
- LARGE_INTEGER [wylączStoper](#) ()
Metoda konczy pomiar czasu.
- void [wykonajAlgorytm](#) ()
Metoda wykonuje algorytm na danych wejscowych (tablicy)
- bool [sprawdz](#) ()
Metoda sprawdza poprawnosc algorytmu.
- int [uruchom](#) (string nazwa)
Metoda wykonuje jednorazowy test algorytmu Metoda: -włącza 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.
- [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 dwuch tablic Scalanie czyli laczenie dwuch posortowanych tablic w jedna(rowniez posortowana)
- [Kontener MergeSort](#) ([Kontener](#) tab)
Metoda implementujca sortowanie przez scalanie.
- int [wez_rozmiar](#) ()

2.1.1 Detailed Description

Klasa modelująca główną część programu.

Klasa modeluje główną część programu, którego zadaniem jest: -wczytanie danych -zmierzenie czasu działania algorytmu -sprawdzenie poprawności tego algorytmu, mając oczekiwany wynik

2.1.2 Member Function Documentation

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

Metoda implementująca sortowanie przez kopcowanie.

Parameters

<i>tab</i>	- tablica do posortowania
------------	---------------------------

Returns

metoda zwraca posortowaną tablicę

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

Metoda implementująca scalanie dwóch tablic Scalanie czyli łączenie dwóch posortowanych tablic w jedną (również posortowaną)

Parameters

<i>lewy</i>	- element pierwszy
<i>prawy</i>	- element drugi

Returns

metoda zwraca scalony zbiór

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

Metoda implementująca sortowanie przez scalanie.

Parameters

<i>tab</i>	- tablica do posortowania
------------	---------------------------

Returns

metoda zwraca posortowaną tablicę

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

Metoda implementująca sortowanie szybkie.

Parameters

<i>tab</i>	- wskaźnik na tablicę do posortowania
<i>lewy</i>	- indeks początku tablicy
<i>prawy</i>	- indeks końca tablicy

2.1.2.5 bool Działanie::sprawdz ()

Metoda sprawdza poprawność algorytmu.

Wczytywane są poprawne dane wynikowe, a następnie są one porównywane z tymi otrzymanymi przez wykonanie algorytmu.

Returns

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

2.1.2.6 void Działanie::uporządkuj_kopiec (Kontener * kopiec)

Metoda przywraca strukturę, jeśli ta została zaburzona.

Parameters

<i>kopiec</i>	- wskaźnik na kopiec do uporządkowania
---------------	--

2.1.2.7 void Działanie::wczytajDaneWejsciowe (string nazwa)

Metoda wczytuje dane do tablicy znajdującej się w zmiennej wejście.

Format danych w pliku jest następujący: pierwszy wiersz - ilość elementów, a następnie w kolumnie kolejne wartości tablicy.

Parameters

<i>nazwaPliku</i>	- nazwa pliku do otwarcia
-------------------	---------------------------

Returns

void

2.1.2.8 LARGE_INTEGER Działanie::włączStoper ()

Metoda uruchamia pomiar czasu.

Czas jest mierzony w milisekundach.

Returns

czas, w którym został włączony stoper

2.1.2.9 void Działanie::wykonajAlgorytm ()

Metoda wykonuje algorytm na danych wejściowych (tablica)

Algorytm do wykonania : pomnoż każdy element razy 2.

Returns

void

2.1.2.10 LARGE_INTEGER Dzialanie::wylaczStoper ()

Metoda konczy pomiar czasu.

Czas jest mierzony w milisekundach

Returns

czas, w którym stoper został wyłączony

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

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

2.2 kolejka_list Class Reference

Klasa modelująca strukturę kolejki Kolejka jest zbudowana w oparciu o listę.

```
#include <kolejka_list.hh>
```

Public Member Functions

- void **enqueue** (int element)
Metoda dodająca element na stos Metoda dodaje element na koniec listy.
- void **dequeue** (int *a)
Metoda usuwająca element ze stos Metoda usuwa element znajdujący się na końcu listy. Gdy stos jest pusty, wyświetlony zostaje błąd mówiący o braku danych do ściągnięcia.
- bool **isempty** ()
Metoda sprawdzająca czy stos jest pusty. Metoda sprawdza czy stos jest pusty.
- int **size** ()
Metoda zwracająca rozmiar stosu Metoda zwraca rozmiar listy.
- void **wyswietl** ()

2.2.1 Detailed Description

Klasa modelująca strukturę kolejki Kolejka jest zbudowana w oparciu o listę.

2.2.2 Member Function Documentation

2.2.2.1 void kolejka_list::dequeue (int * a)

Metoda usuwająca element ze stos Metoda usuwa element znajdujący się na końcu listy. Gdy stos jest pusty, wyświetlony zostaje błąd mówiący o braku danych do ściągnięcia.

Parameters

a	- wskaźnik do zmiennej, do której ściągamy wartość.
----------	---

2.2.2.2 void kolejka_list::enqueue (int element)

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

Parameters

<i>element</i>	- element, który zostanie dodany do stosu
----------------	---

2.2.2.3 bool kolejka_list::isempty ()

Metoda sprawdzająca czy stos jest pusty. Metoda sprawdza czy stos jest pusty.

Returns

lista.empty();

2.2.2.4 int kolejka_list::size ()

Metoda zwracająca 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 modelująca strukturę kolejki Kolejka jest zbudowana w oparciu o listę.

```
#include <kolejka_tab.hh>
```

Public Member Functions

- **kolejka_tab** (int zwiększ)
- void **enqueue** (int element)
Metoda dodająca element na stos Metoda dodaje element na koniec listy.
- void **dequeue** (int *a)
Metoda usuwająca element ze stosu Metoda usuwa element znajdujący się na końcu listy. Gdy stos jest pusty, wyświetlony zostaje błąd mówiący o braku danych do ściągnięcia.
- bool **isempty** ()
Metoda sprawdzająca czy stos jest pusty. Metoda sprawdza czy stos jest pusty.
- int **size** ()
Metoda zwracająca rozmiar stosu Metoda zwraca rozmiar listy.
- void **wyswietl** ()

2.3.1 Detailed Description

Klasa modelująca strukturę kolejki Kolejka jest zbudowana w oparciu o listę.

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

<i>a</i>	- 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

<i>element</i>	- 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 wypisywania Metoda pozwala na wypisanie zawartosci tablicy na standardowe wyjście.

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

<i>e</i>	- wartosc elementu
----------	--------------------

2.4.2.2 void Kontener::dodaj_elementy (Kontener tab)

Metoda dodaje na koniec tablicy zawartosc innej tablicy.

Parameters

<i>tablica,ktora</i>	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

<i>tablica, ktora</i>	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

<i>tablica, z</i>	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

<i>index</i>	- indeks, ktorego referencja zostanie zwrzcona
--------------	--

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

<i>nazwaPliku</i>	- nazwa pliku do otwarcia
-------------------	---------------------------

Returns

void

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

Metoda zwracająca referencje do tablicy danych Metoda pozwala na dostęp do tablicy i jej modyfikacje.

Returns

referencja do tablicy

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

Metoda zwracająca rozmiar tablicy.

Returns

rozmiar tablicy

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

Metoda zamienia ze sobą dwa elementy tablicy.

Parameters

<i>i,j</i>	- indeksy, które zostaną zamienione
------------	-------------------------------------

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

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

2.5 stos_list Class Reference

Klasa modelująca strukturę stosu Stos jest zbudowany w oparciu o listę.

```
#include <stos_list.hh>
```

Public Member Functions

- void `push` (int element)
Metoda dodająca element na stos Metoda dodaje element na koniec listy.
- void `pop` (int *a)
Metoda usuwająca element ze stosu Metoda usuwa element znajdujący się na końcu listy. Gdy stos jest pusty, wyświetlony zostaje błąd mówiący o braku danych do ściągnięcia.
- bool `isempty` ()
Metoda sprawdzająca czy stos jest pusty. Metoda sprawdza czy stos jest pusty.

- int `size` ()

Metoda zwracajaca rozmiar stosu Metoda zwraca rozmiar listy.

- void `wyswietl` ()

2.5.1 Detailed Description

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

2.5.2 Member Function Documentation

2.5.2.1 bool `stos_list::isempty` ()

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

Returns

`lista.empty();`

2.5.2.2 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

<i>a</i>	- wskaznik do zmiennej, do ktorej sciagamy wartosc.
----------	---

2.5.2.3 void `stos_list::push` (int *element*)

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

Parameters

<i>element</i>	- element, ktory zostanie dodany do stosu
----------------	---

2.5.2.4 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`

2.6 `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 *zwiększ*)
- void **push** (int *element*)
Metoda dodająca element na stos Metoda dodaje element na koniec tablicy. W przypadku gdy tablica jest pełna zostaje ona powiększona o 1 element.
- void **pop** (int **a*)
Metoda usuwająca element ze stos Metoda usuwa element znajdujący się na końcu tablicy. Z każdorazowym pobraniem danych rozmiar tablicy jest zmniejszany o 1.
- bool **isempty** ()
Metoda sprawdzająca czy stos jest pusty. Metoda sprawdza czy stos jest pusty. Sprawdzenie polega na odczytaniu zmiennej rozmiar, która pamięta ile jest elementów na stosie.
- int **size** ()
Metoda zwracająca rozmiar stosu Metoda zwraca rozmiar stosu. Rozmiar stosu przechowywany jest w zmiennej rozmiar.
- void **wyswietl** ()

2.6.1 Detailed Description

Klasa modelująca strukturę stosu Stos jest zbudowany w oparciu o tablice.

2.6.2 Member Function Documentation

2.6.2.1 bool stos_tab::isempty ()

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

która pamięta ile jest elementów na stosie.

Returns

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

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

Metoda usuwająca element ze stos Metoda usuwa element znajdujący się na końcu tablicy. Z każdorazowym pobraniem danych rozmiar tablicy jest zmniejszany o 1.

Parameters

<i>a</i>	- wskaźnik do zmiennej, do której ściągamy wartość.
----------	---

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

Metoda dodająca element na stos Metoda dodaje element na koniec tablicy. W przypadku gdy tablica jest pełna zostaje ona powiększona o 1 element.

Parameters

<i>element</i>	- element, który zostanie dodany do stosu
----------------	---

2.6.2.4 int stos_tab::size ()

Metoda zwracająca 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

2.7 Tester Class Reference

Klasa [Tester](#) Klasa modeluje narzędzie do benchmarkowania algorytmu. Poprzez wielokrotne wynonywanie algorytmu możemy wyciągnąć średni czas działania, co jest dokładniejszym pomiarem.

```
#include <tester.hh>
```

Public Member Functions

- [Tester](#) ()
konstruktor klasy Konstruktor inicjuje wartości: -powtorzenia (ile razy wykonywać algorytm) -ilosc (ile mamy zestawów danych) -wejście (nazwa pliku z danymi wejściowymi) -wynik (nazwa pliku z poprawnym wynikiem algorytmu)
- void [otworzPlik](#) (string nazwa)
Metoda otwierająca referencje do pliku CSV.
- void [zamknijPlik](#) ()
Metoda zamykająca referencje do pliku CSV.
- void [symulacja](#) ()
Metoda symulująca badanie algorytmu Metoda wykonuje symulację działania algorytmu. Wykonuje algorytm zadana liczba razy dla zadanych zestawów danych. Wyniki zapisuje do pliku CSV o nazwie rezultat.csv Format zapisu: rozmiar_problemu,ilosc_powtorzen,średni_czas.
- void [zamienNazwy](#) (int numer)
Metoda aktualizuje nazwę plików wejściowych Format plików wejściowych jest ściśle określony. Dane wejściowe: wejście%.txt, gdzie % to numer zestawu Dane wynikowe: wynik%.txt, gdzie % to numer zestawu Zestawy należy numerować od 0.

2.7.1 Detailed Description

Klasa [Tester](#) Klasa modeluje narzędzie do benchmarkowania algorytmu. Poprzez wielokrotne wynonywanie algorytmu możemy wyciągnąć średni czas działania, co jest dokładniejszym pomiarem.

2.7.2 Member Function Documentation

2.7.2.1 void Tester::zamienNazwy (int numer)

Metoda aktualizuje nazwę plików wejściowych Format plików wejściowych jest ściśle określony. Dane wejściowe: wejście%.txt, gdzie % to numer zestawu Dane wynikowe: wynik%.txt, gdzie % to numer zestawu Zestawy należy numerować od 0.

Parameters

<i>numer</i>	- % zostaje zamieniony na wartość 'numer'
--------------	---

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

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

Index

- dequeue
 - kolejka_list, 6
 - kolejka_tab, 8
- dodaj_element
 - Kontener, 9
- dodaj_elementy
 - Kontener, 9
- Dzialanie, 3
 - heap, 4
 - merge, 4
 - MergeSort, 4
 - Quicksort, 4
 - sprawdz, 5
 - uporzadkuj_kopiec, 5
 - wczytajDaneWejscowe, 5
 - wlaczStoper, 5
 - wykonajAlgorytm, 5
 - wylaczStoper, 5
- enqueue
 - kolejka_list, 6
 - kolejka_tab, 8
- heap
 - Dzialanie, 4
- isempty
 - kolejka_list, 7
 - kolejka_tab, 8
 - stos_list, 12
 - stos_tab, 13
- kolejka_list, 6
 - dequeue, 6
 - enqueue, 6
 - isempty, 7
 - size, 7
- kolejka_tab, 7
 - dequeue, 8
 - enqueue, 8
 - isempty, 8
 - size, 8
- Kontener, 8
 - dodaj_element, 9
 - dodaj_elementy, 9
 - operator+, 9
 - operator=, 10
 - operator==, 10
 - wczytajDane, 10
 - wez_dane, 11
 - wez_rozmiar, 11
 - zamien_elementy, 11
- merge
 - Dzialanie, 4
- MergeSort
 - Dzialanie, 4
- operator+
 - Kontener, 9
- operator=
 - Kontener, 10
- operator==
 - Kontener, 10
- pop
 - stos_list, 12
 - stos_tab, 13
- push
 - stos_list, 12
 - stos_tab, 13
- Quicksort
 - Dzialanie, 4
- size
 - kolejka_list, 7
 - kolejka_tab, 8
 - stos_list, 12
 - stos_tab, 13
- sprawdz
 - Dzialanie, 5
- stos_list, 11
 - isempty, 12
 - pop, 12
 - push, 12
 - size, 12
- stos_tab, 12
 - isempty, 13
 - pop, 13
 - push, 13
 - size, 13
- Tester, 14
 - zamienNazwy, 14
- uporzadkuj_kopiec
 - Dzialanie, 5
- wczytajDane
 - Kontener, 10

wczytajDaneWejscowe
 Dzialanie, [5](#)
wez_dane
 Kontener, [11](#)
wez_rozmiar
 Kontener, [11](#)
włączStoper
 Dzialanie, [5](#)
wykonajAlgorytm
 Dzialanie, [5](#)
wylaczStoper
 Dzialanie, [5](#)

zamien_elementy
 Kontener, [11](#)
zamienNazwy
 Tester, [14](#)