ADT

0.1

Generated by Doxygen 1.8.6

Mon Apr 11 2016 02:18:28

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

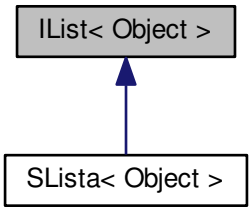
Class Documentation

4.1 IList< Object > Class Template Reference

Interfejs listy jednokierunkowej.

`#include <IList.hh>`

Inheritance diagram for IList< Object >:



Public Member Functions

- virtual bool IsEmpty ()=0

  Metoda sprawdzająca, czy lista jest pusta.
- virtual const Object & Front ()=0

  Metoda zwracająca pierwszy element listy.
- virtual void AddFront (const Object newItem)=0

  Metoda dodająca element na początek listy.
- virtual void RemoveFront ()=0

  Metoda usuwająca element z początku listy.

4.1.1 Detailed Description

template<typename Object>class IList< Object >

Definiuje ADT dla listy jednokierunkowej.

Lista może przechowywać dowolny typ danych dzięki zastosowaniu szablonu.

Definition at line 22 of file IList.hh.

4.1.2 Member Function Documentation

4.1.2.1 template<typename Object> virtual void IList< Object >::AddFront (const Object *newItem*) [pure virtual]

Parameters

in	*newItem*	- element do dodania

Implemented in SLista< Object >, and SLista< int >.

4.1.2.2 template<typename Object> virtual const Object& IList< Object >::Front ( ) [pure virtual]

Returns

pierwszy element listy

Implemented in SLista< Object >, and SLista< int >.

4.1.2.3 template<typename Object> virtual bool IList< Object >::IsEmpty ( ) [pure virtual]

true - jeśli lista jest pusta truefalse - jeśli nie jest pusta

Implemented in SLista< Object >, and SLista< int >.

4.1.2.4 template<typename Object> virtual void IList< Object >::RemoveFront ( ) [pure virtual]

Implemented in SLista< Object >, and SLista< int >.

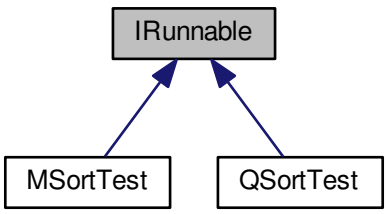
The documentation for this class was generated from the following file:

- IList.hh

4.2 IRunnable Class Reference

`#include <IRunnable.hh>`

Inheritance diagram for IRunnable:



Public Member Functions

- virtual void Przygotuj ()=0
- virtual void Testuj ()=0

4.2.1 Detailed Description

Definition at line 12 of file IRunnable.hh.

4.2.2 Member Function Documentation

4.2.2.1 virtual void IRunnable::Przygotuj ( ) [pure virtual]

Implemented in MSortTest, and QSortTest.

4.2.2.2 virtual void IRunnable::Testuj ( ) [pure virtual]

Implemented in MSortTest, and QSortTest.

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

- IRunnable.hh

4.3 IStack< Object > Class Template Reference

Klasa szablonowa modelująca interfejs stosu.

`#include <IStack.hh>`

Public Member Functions

- virtual bool IsEmpty ()=0

  Metoda sprawdzająca, czy stos jest pusty.
- virtual int Size ()=0

  Metoda obliczająca rozmiar stosu.
- virtual Object & Top ()=0

  Metoda zwracająca wierzchołek stosu.
- virtual void Push (Object item)=0

  Metoda dodająca element na stos.
- virtual Object & Pop ()=0

  Metoda zrzucająca element ze stosu.

4.3.1 Detailed Description

template<typename Object>class IStack< Object >

Definiuje ADT dla stosu.

Stos może przechowywać dowolny typ danych dzięki zastosowaniu szablonu.

Definition at line 22 of file IStack.hh.

4.3.2 Member Function Documentation

4.3.2.1 template<typename Object > virtual bool IStack< Object >::IsEmpty ( ) [pure virtual]

true - jeśli stos jest pustay false - jeśli stos nie jest pusty

4.3.2.2 template<typename Object > virtual Object& IStack< Object >::Pop ( ) [pure virtual]

Usuwa wierzchołek ze stosu i zwraca jego wartość.

Returns

element na wierzchu stosu

4.3.2.3 template<typename Object > virtual void IStack< Object >::Push (Object *item*) [pure virtual]

Wrzuca element na wierzchołek stosu.

Parameters

in	*item*	- element do dodania

4.3.2.4 template<typename Object > virtual int IStack< Object >::Size ( ) [pure virtual]

Returns

liczba elementów na stos

4.3.2.5 template<typename Object > virtual Object& IStack< Object >::Top ( ) [pure virtual]

Returns

element na wierzchu stosu

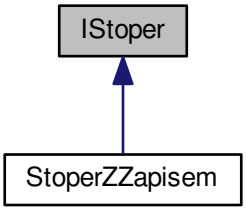
The documentation for this class was generated from the following file:

- IStack.hh

4.4 IStoper Class Reference

`#include <IStoper.hh>`

Inheritance diagram for IStoper:



Public Member Functions

- virtual void Start ()
- virtual void Stop ()
- virtual double GetElapsedTime ()

Protected Attributes

- timeval start
- timeval stop

4.4.1 Detailed Description

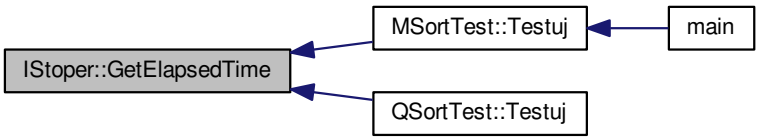
Definition at line 9 of file IStoper.hh.

4.4.2 Member Function Documentation

4.4.2.1 double IStoper::GetElapsedTime ( ) [virtual]

Definition at line 12 of file IStoper.cpp.

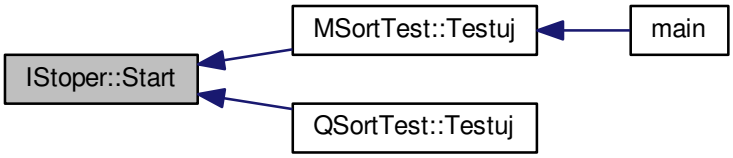
Here is the caller graph for this function:



4.4.2.2 void IStoper::Start ( ) [virtual]

Definition at line 4 of file IStoper.cpp.

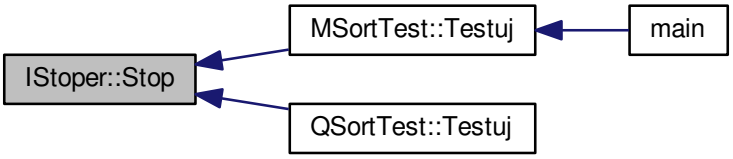
Here is the caller graph for this function:



4.4.2.3 void IStoper::Stop ( ) [virtual]

Definition at line 8 of file IStoper.cpp.

Here is the caller graph for this function:



4.4.3 Member Data Documentation

4.4.3.1 timeval IStoper::start [protected]

Definition at line 12 of file IStoper.hh.

4.4.3.2 timeval IStoper::stop [protected]

Definition at line 12 of file IStoper.hh.

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

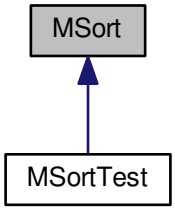
- IStoper.hh
- IStoper.cpp

4.5 MSort Class Reference

Merge Sort.

`#include <MSort.hh>`

Inheritance diagram for MSort:



Public Member Functions

- void Scal (Tab &tablica, int lewy, int srodek, int prawy)

  Funkcja scalająca dwie "podtablice" w jedna.
- void Mergesort (Tab &tablica, int lewy, int prawy)

  Mergesort na tablicy.

4.5.1 Detailed Description

Definition at line 20 of file MSort.hh.

4.5.2 Member Function Documentation

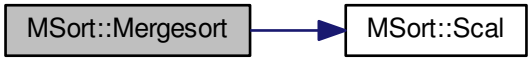
4.5.2.1 void MSort::Mergesort (Tab & *tablica,* int *lewy,* int *prawy*)

Parameters

in	*tablica-struktura*	przechowujaca tablice dynamiczna
in	*lewy-*	lewa granica sortowania (numer indeksu)
in	*prawy-*	prawa granica sortowania (numer indeksu)

Definition at line 40 of file MSort.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

4.5.2.2 void MSort::Scal (Tab & *tablica*, int *lewy*, int *srodek*, int *prawy*)

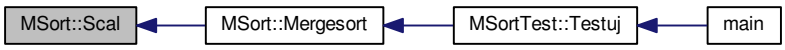
Funkcja przepisuje dwie "podtablice" do jednej zapewniajac przy tym juz posortowane ulozenie elementow

Parameters

in	*tablica-struktura*	przechowujaca tablice dynamiczna
in	*lewy-*	lewa granica pierwszej podtablicy
in	*srodek-*	prawa granica pierwszej podtablicy
in	*prawy-*	prawa granica drugiej podtablicy

Definition at line 14 of file MSort.cpp.

Here is the caller graph for this function:



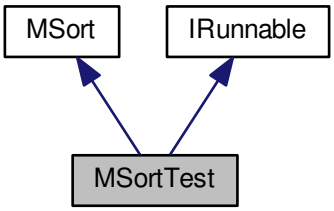
The documentation for this class was generated from the following files:

- MSort.hh
- MSort.cpp

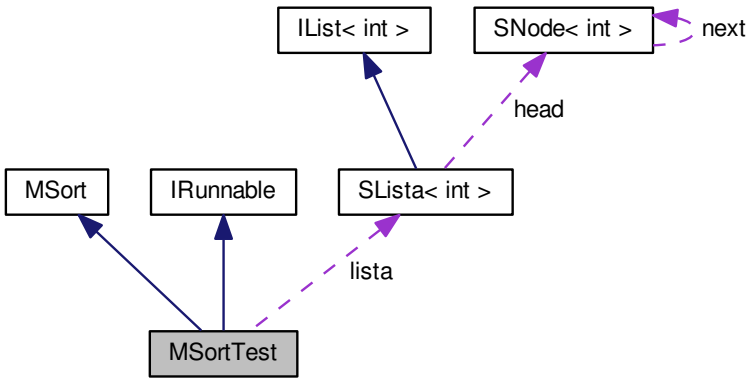
4.6 MSortTest Class Reference

`#include <MSortTest.hh>`

Inheritance diagram for MSortTest:



Collaboration diagram for MSortTest:



Public Member Functions

- void Przygotuj ()
- void Testuj ()

Private Attributes

- SLista< int > lista

4.6.1 Detailed Description

Definition at line 17 of file MSortTest.hh.

4.6.2 Member Function Documentation

4.6.2.1 void MSortTest::Przygotuj ( ) [virtual]

Uzupelnia liste liczbami oznaczajacymi ile elementow ma byc zapisane do tablicy w procesie testowania

Implements IRunnable.

Definition at line 7 of file MSortTest.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

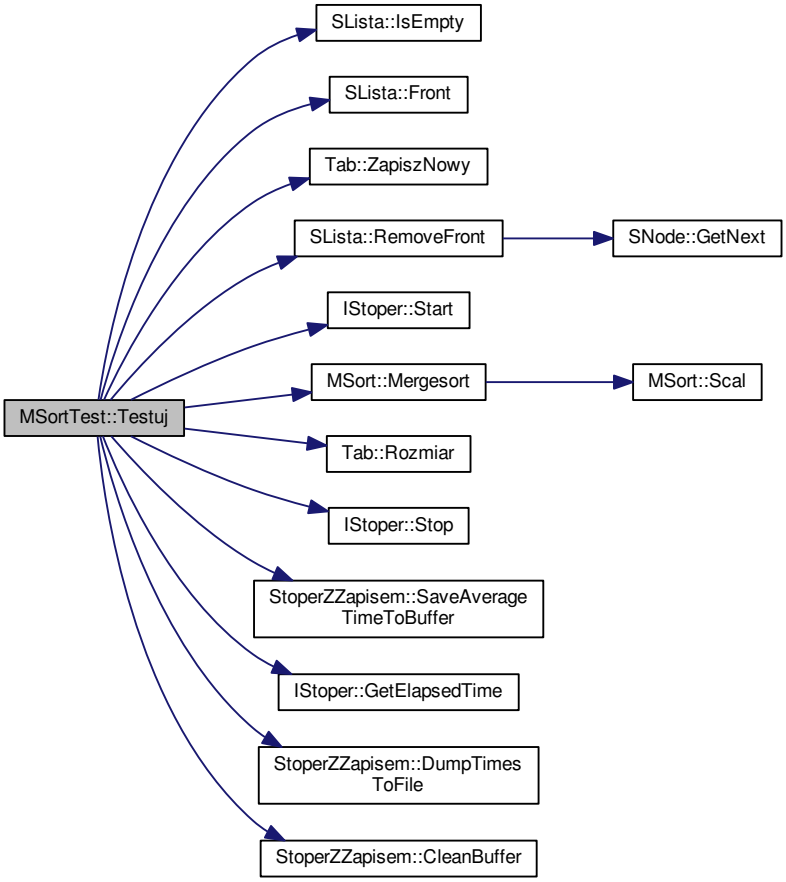
4.6.2.2 void MSortTest::Testuj ( ) [virtual]

Wypelnia tablice i sortuje jej elementy

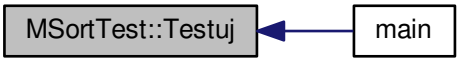
Implements IRunnable.

Definition at line 22 of file MSortTest.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



4.6.3 Member Data Documentation

4.6.3.1 SLista<int> MSortTest::lista [private]

Definition at line 18 of file MSortTest.hh.

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

- MSortTest.hh

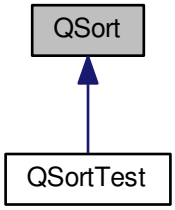
- MSortTest.cpp

4.7 QSort Class Reference

QuickSort.

`#include <QSort.hh>`

Inheritance diagram for QSort:



Public Member Functions

- void Quicksort (Tab &tablica, int lewy, int prawy)

  QuickSort na tablicy (pivot srodkowym elementem tablicy)
- void QuicksortLewy (Tab &tablica, int lewy, int prawy)

  QuickSort na tablicy (pivot pierwszym elementem tablicy)
- int Maksimum (Tab &tablica, int lewy, int prawy)

  pomocnicza funkcja dla nieuzywanej przy testowaniu funkcji "QuicksortPesym
- void QuicksortPesym (Tab &tablica, int lewy, int prawy)

  QuickSort na tablicy (nieuzywane przy testowaniu)

4.7.1 Detailed Description

Definition at line 19 of file QSort.hh.

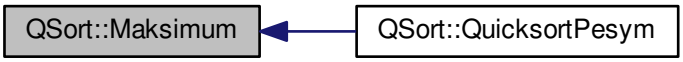
4.7.2 Member Function Documentation

4.7.2.1 int QSort::Maksimum (Tab & *tablica,* int *lewy,* int *prawy*)

Znajduje maksymalna wartosc w tablicy i zwraca indeks

Definition at line 65 of file QSort.cpp.

Here is the caller graph for this function:



4.7.2.2 void QSort::Quicksort (Tab & *tablica,* int *lewy,* int *prawy*)

Parameters

in	*tablica-struktura*	przechowujaca tablice dynamiczna
in	*lewy-*	lewa granica sortowania (numer indeksu)
in	*prawy-*	prawa granica sortowania (numer indeksu)

Definition at line 10 of file QSort.cpp.

Here is the caller graph for this function:

4.7.2.3 void QSort::QuicksortLewy (Tab & *tablica,* int *lewy,* int *prawy*)

Parameters

in	*tablica-struktura*	przechowujaca tablice dynamiczna
in	*lewy-*	lewa granica sortowania (numer indeksu)
in	*prawy-*	prawa granica sortowania (numer indeksu)

Definition at line 38 of file QSort.cpp.

4.7.2.4 void QSort::QuicksortPesym (Tab & *tablica,* int *lewy,* int *prawy*)

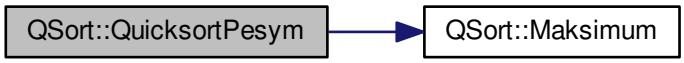
Sztucznie wytworzony przypadek pesymistyczny dla dowolnego zestawu danych elementem rozdzielającym za każdym razem okazuje się element podtablicy T o skrajnej wartości (maksymalnej lub minimalnej) UWAGA: pomiar czasu dodatkowo obciazony jest funkcja "Maksimum"

Parameters

in	*tablica-struktura*	przechowujaca tablice dynamiczna
in	*lewy-*	lewa granica sortowania (numer indeksu)
in	*prawy-*	prawa granica sortowania (numer indeksu)

Definition at line 86 of file QSort.cpp.

Here is the call graph for this function:



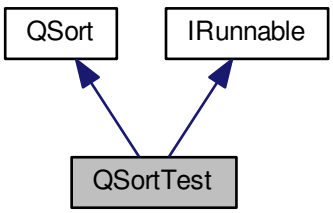
The documentation for this class was generated from the following files:

- QSort.hh
- QSort.cpp

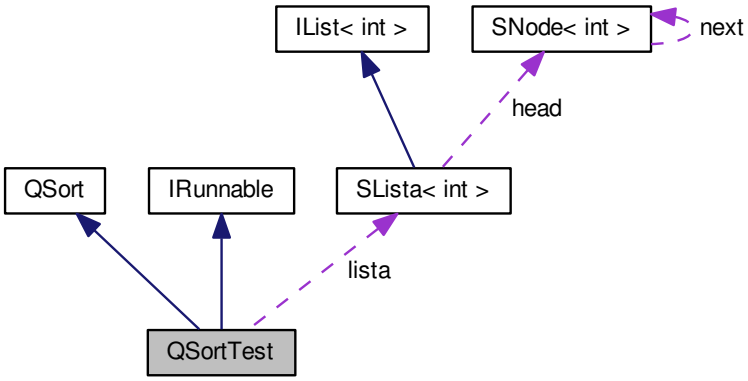
4.8 QSortTest Class Reference

`#include <QSortTest.hh>`

Inheritance diagram for QSortTest:



Collaboration diagram for QSortTest:



Public Member Functions

- void Przygotuj ()
- void Testuj ()

Private Attributes

- SLista< int > lista

4.8.1 Detailed Description

Definition at line 17 of file QSortTest.hh.

4.8.2 Member Function Documentation

4.8.2.1 void QSortTest::Przygotuj ( ) [virtual]

Uzupelnia liste liczbami oznaczajacymi ile elementow ma byc zapisane do tablicy w procesie testowania

Implements IRunnable.

Definition at line 7 of file QSortTest.cpp.

Here is the call graph for this function:

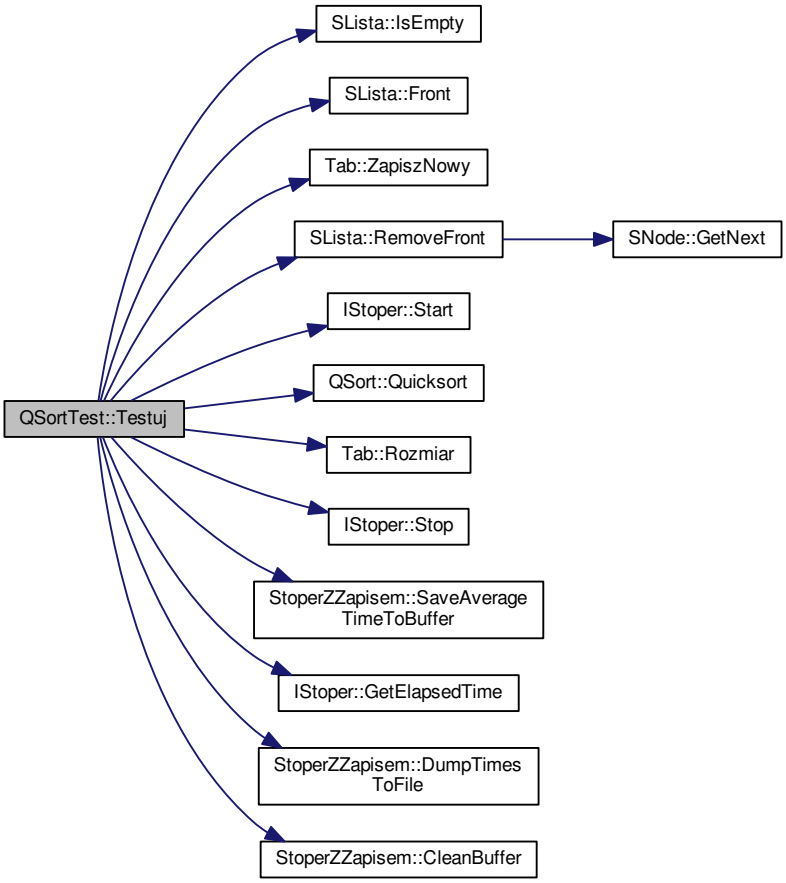
4.8.2.2 void QSortTest::Testuj ( ) [virtual]

Wypelnia tablice i sortuje jej elementy

Implements IRunnable.

Definition at line 22 of file QSortTest.cpp.

Here is the call graph for this function:



4.8.3 Member Data Documentation

4.8.3.1 SLista<int> QSortTest::lista [private]

Definition at line 18 of file QSortTest.hh.

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

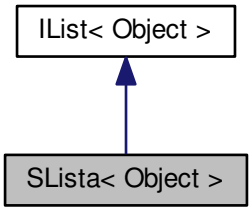
- QSortTest.hh
- QSortTest.cpp

4.9 SLista< Object > Class Template Reference

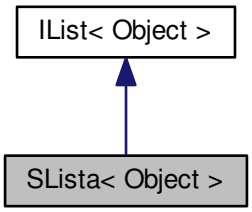
Szablonowa klasa implementująca listę jednokierunkową

`#include <SLista.hh>`

Inheritance diagram for SLista< Object >:



Collaboration diagram for SLista< Object >:



Public Member Functions

- SLista ()
  Konstruktor listy jednokierunkowaj.
- ~SLista ()
  Destruktor listy jednokierunkowaj.
- SNode< Object > * Head ()
  Metoda zwracająca głowę listy.
- SNode< Object > * Find (Object k)
  Metoda wyszukująca element na liście.
- virtual bool IsEmpty ()
  Metoda sprawdzająca, czy lista jest pusta.
- virtual const Object & Front ()
  Metoda zwracająca pierwszy element listy.
- virtual void AddFront (const Object newItem)
  Metoda dodająca element na początek listy.
- virtual void RemoveFront ()
  Metoda usuwająca element z początku listy.
- void printl ()

Private Attributes

- SNode< Object > * head

4.9.1 Detailed Description

template<typename Object>class SLista< Object >

SLista jest zbudowana w oparciu o węzły SNode oraz operacje na wskaźnikach.

SLista może przechowywać dowolny typ danych dzięki zastosowaniu szablonu.

Definition at line 25 of file SLista.hh.

4.9.2 Constructor & Destructor Documentation

4.9.2.1 template<typename Object > SLista< Object >::SLista ( )

Inicjuje SListę poprzez ustawienie wskaźnika NULL jako początek (head) tej listy.

Definition at line 127 of file SLista.hh.

4.9.2.2 template<typename Object > SLista< Object >::~SLista ( )

Usuwa SListę poprzez ustawienie wskaźnika NULL jako początek (head) tej listy.

Definition at line 131 of file SLista.hh.

4.9.3 Member Function Documentation

4.9.3.1 template<typename Object> void SLista< Object >::AddFront (const Object *newItem*) [virtual]

Alokuje nowy węzeł, dodaje nowy element, dodaje powiązanie tak, aby węzeł wskazywał na stary head, uaktualnia head.

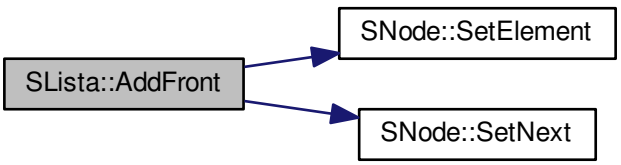
Parameters

in	*newItem*	- element do dodania

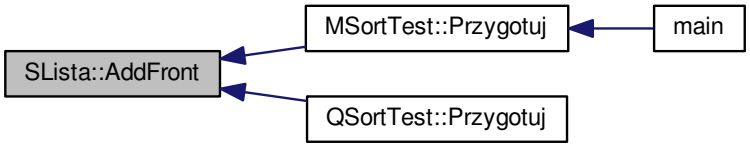
Implements IList< Object >.

Definition at line 165 of file SLista.hh.

Here is the call graph for this function:



Here is the caller graph for this function:



4.9.3.2 template<typename Object> SNode< Object > * SLista< Object >::Find (Object *k*)

Implementuje algorytm liniowego przeszukiwania listy.

Parameters

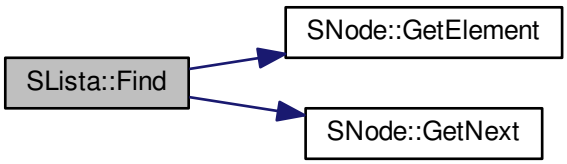
in	*k*	- element do wyszukania

Returns

Wskaźnik do znalezionego elementu lub NULL, gdy nie znaleziono.

Definition at line 117 of file SLista.hh.

Here is the call graph for this function:



4.9.3.3 template<typename Object > const Object & SLista< Object >::Front ( ) [virtual]

Sprawdza, czy lista jest pusta i zwraca dane pierwszego węzła listy. Jeśli lista jest pusta, wyrzuca wyjątek.

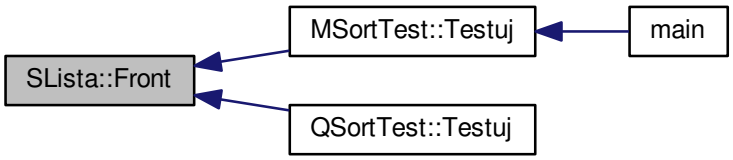
Returns

dane pierwszego węzła listy

Implements IList< Object >.

Definition at line 156 of file SLista.hh.

Here is the caller graph for this function:



4.9.3.4 template<typename Object > SNode< Object > * SLista< Object >::Head ( )

Zwraca wskaźnik do początku listy lub NULL, jeśli lista jest pusta.

Returns

Wskaźnik do głowy listy.

Definition at line 147 of file SLista.hh.

4.9.3.5 template<typename Object > bool SLista< Object >::IsEmpty ( ) [virtual]

Sprawdza, czy head wskazuje na coś innego niż NULL. Implementacja metody wirtualnej z interfejsu IList.

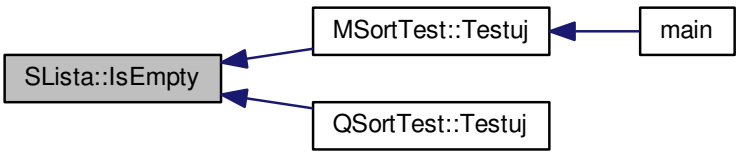
Returns

true - jeśli lista jest pusta, false - jeśli nie

Implements IList< Object >.

Definition at line 138 of file SLista.hh.

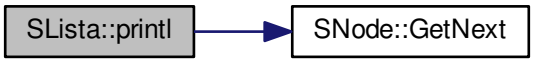
Here is the caller graph for this function:



4.9.3.6 template<typename Object > void SLista< Object >::printl ( )

Definition at line 109 of file SLista.hh.

Here is the call graph for this function:



4.9.3.7 template<typename Object > void SLista< Object >::RemoveFront ( ) [virtual]

Uaktualnia head, aby wskazywał na kolejny element na liście, po czym usuwa stary węzeł.

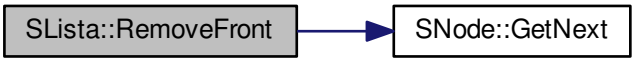
Parameters

in	*newItem*	- element do dodania

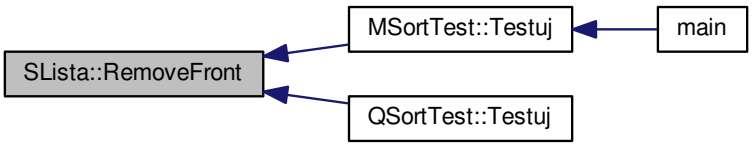
Implements IList< Object >.

Definition at line 174 of file SLista.hh.

Here is the call graph for this function:



Here is the caller graph for this function:



4.9.4 Member Data Documentation

4.9.4.1 template<typename Object> SNode<Object>* SLista< Object >::head [private]

Definition at line 28 of file SLista.hh.

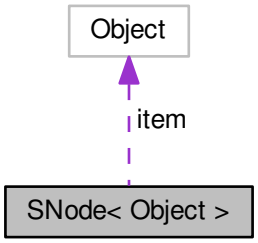
The documentation for this class was generated from the following file:

- SLista.hh

4.10 SNode< Object > Class Template Reference

`#include <SNode.hh>`

Collaboration diagram for SNode< Object >:



Public Member Functions

- Object GetElement ()
- SNode< Object > * GetNext ()
- void SetElement (Object newItem)
- void SetNext (SNode< Object > *newItem)

Private Attributes

- Object item
- SNode< Object > * next

4.10.1 Detailed Description

template<typename Object>class SNode< Object >

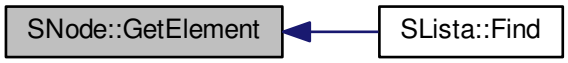
Definition at line 8 of file SNode.hh.

4.10.2 Member Function Documentation

4.10.2.1 template<typename Object> Object SNode< Object >::GetElement ( ) [inline]

Definition at line 14 of file SNode.hh.

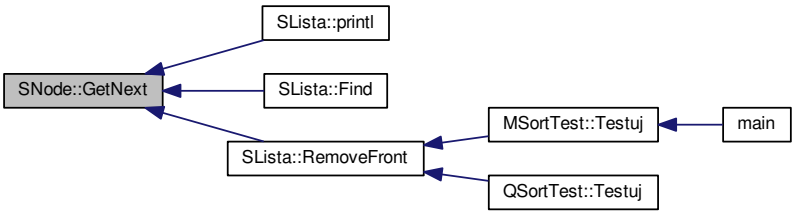
Here is the caller graph for this function:



4.10.2.2 template<typename Object> SNode<Object>* SNode< Object >::GetNext ( ) [inline]

Definition at line 15 of file SNode.hh.

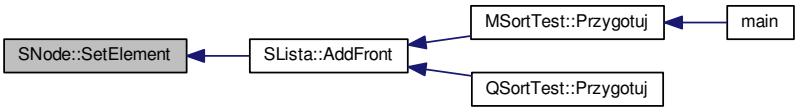
Here is the caller graph for this function:



4.10.2.3 template<typename Object> void SNode< Object >::SetElement (Object *newItem*) [inline]

Definition at line 16 of file SNode.hh.

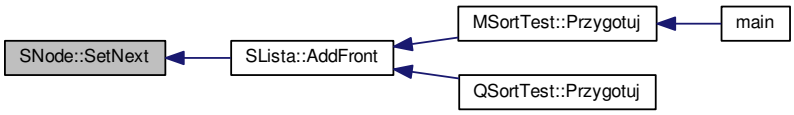
Here is the caller graph for this function:



4.10.2.4 template<typename Object> void SNode< Object >::SetNext (SNode< Object > * *newItem*) [inline]

Definition at line 17 of file SNode.hh.

Here is the caller graph for this function:



4.10.3 Member Data Documentation

4.10.3.1 template<typename Object> Object SNode< Object >::item [private]

Definition at line 11 of file SNode.hh.

4.10.3.2 template<typename Object> SNode<Object>* SNode< Object >::next [private]

Definition at line 12 of file SNode.hh.

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

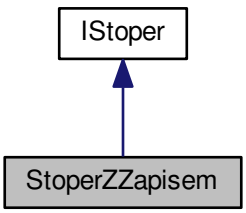
- SNode.hh

4.11 StoperZZapisem Class Reference

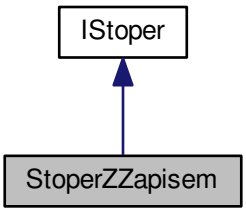
Klasa implementująca rozbudowany stoper.

`#include <StoperZZapisem.hh>`

Inheritance diagram for StoperZZapisem:



Collaboration diagram for StoperZZapisem:



Public Member Functions

- StoperZZapisem ()
- ~StoperZZapisem ()
- int & Rozmiar ()
- double *& Poczatek ()
- double *& PoczatekBufora ()
- bool SaveElapsedTime (double rekord)

  Metoda zapisująca wartość pomiaru czasu okrążenia.
- bool SaveAverageTimeToBuffer (double rekord)

  Metoda zapisująca średni czas okrążenia do bufora plikowego.
- void ShowMemory ()

  Metoda wypisująca zawartość pamięci stopera.
- void CleanMemory ()

  Metoda usuwająca zawartość pamięci stopera.
- void CleanBuffer ()

  Metoda usuwająca zawartość bufora plikowego stopera.
- double SeriesAverage ()

  Metoda wyliczająca średni czas okrążenia.
- bool DumpTimesToFile (ofstream &plik)

  Metoda zapisująca zawartość bufora plikowego do pliku.
- bool DumpOneTimeToFile (ofstream &plik, double rekord)

  Metoda zapisująca pojedynczy rekord bufora plikowego do pliku.

Private Attributes

- int rozm = 0
- double * Notatnik
- double * DoPliku

Additional Inherited Members

4.11.1 Detailed Description

Klasa jest modelelem stopera z funkcją zapisu czasu okrążeń, liczeniem średniego czasu kilku okrążeń, zapisu zmierzonych czasów do pliku.

Definition at line 26 of file StoperZZapisem.hh.

4.11.2 Constructor & Destructor Documentation

4.11.2.1 StoperZZapisem::StoperZZapisem ( )

Definition at line 4 of file StoperZZapisem.cpp.

4.11.2.2 StoperZZapisem::~StoperZZapisem ( ) [inline]

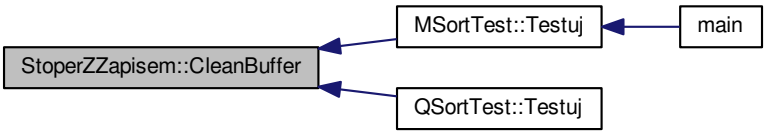
Definition at line 33 of file StoperZZapisem.hh.

4.11.3 Member Function Documentation

4.11.3.1 void StoperZZapisem::CleanBuffer ( )

Definition at line 49 of file StoperZZapisem.cpp.

Here is the caller graph for this function:



4.11.3.2 void StoperZZapisem::CleanMemory ( )

Definition at line 42 of file StoperZZapisem.cpp.

4.11.3.3 bool StoperZZapisem::DumpOneTimeToFile (ofstream & *plik,* double *rekord*)

Dokonuje zapisu wybranego rekordu w buforze do pliku.

Parameters

in	*plik*	- dowiązanie do pliku, do którego wykona zapis
in	*rekord*	- wartość pomiaru czasu, która ma być zapisana

Return values

true	- jeśli udało się zapisać
false	- jeśli udało się zapisać

Definition at line 83 of file StoperZZapisem.cpp.

4.11.3.4 bool StoperZZapisem::DumpTimesToFile (ofstream & *plik*)

Dokonuje zapisu rekordów w buforze do pliku.

Parameters

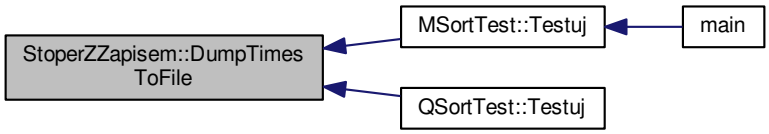
in	*plik*	- dowiązanie do pliku, do którego wykona zapis

Return values

true	- jeśli udało się zapisać
false	- jeśli udało się zapisać

Definition at line 68 of file StoperZZapisem.cpp.

Here is the caller graph for this function:



4.11.3.5 double* & StoperZZapisem::Poczatek ( ) [inline]

Definition at line 35 of file StoperZZapisem.hh.

4.11.3.6 double* & StoperZZapisem::PoczatekBufora ( ) [inline]

Definition at line 36 of file StoperZZapisem.hh.

4.11.3.7 int& StoperZZapisem::Rozmiar ( ) [inline]

Definition at line 34 of file StoperZZapisem.hh.

4.11.3.8 bool StoperZZapisem::SaveAverageTimeToBuffer (double *rekord*)

Dodaje podany czas do pamięci stopera, z której można dokonać zapisu do pliku.

Parameters

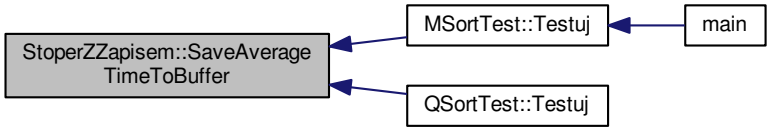
in	*rekord*	- wartość pomiaru czasu

Return values

true	- jeśli udało się zapisać
false	- jeśli udało się zapisać

Definition at line 26 of file StoperZZapisem.cpp.

Here is the caller graph for this function:



4.11.3.9 bool StoperZZapisem::SaveElapsedTime (double *rekord*)

Dodaje podany czas do tablicy czasów okrążeń.

Parameters

in	*rekord*	- wartość pomiaru czasu

Return values

true	- jeśli udało się zapisać
false	- jeśli udało się zapisać

Definition at line 15 of file StoperZZapisem.cpp.

4.11.3.10 double StoperZZapisem::SeriesAverage ( )

Definition at line 55 of file StoperZZapisem.cpp.

4.11.3.11 void StoperZZapisem::ShowMemory ( )

Definition at line 35 of file StoperZZapisem.cpp.

4.11.4 Member Data Documentation

4.11.4.1 double* StoperZZapisem::DoPliku [private]

Definition at line 30 of file StoperZZapisem.hh.

4.11.4.2 double* StoperZZapisem::Notatnik [private]

Definition at line 29 of file StoperZZapisem.hh.

4.11.4.3 int StoperZZapisem::rozm = 0 [private]

Definition at line 28 of file StoperZZapisem.hh.

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

- StoperZZapisem.hh
- StoperZZapisem.cpp

4.12 Tab Class Reference

`#include <Tab.hh>`

Public Member Functions

- Tab ()
- ~Tab ()
- int & Pojemnosc ()
- int & Rozmiar ()
- int *& Poczatek ()
- void WypelnijTab ()
- void WypiszTab ()
- int * WypelnijOdNowa ()
- void ZapiszNowy (int elem)
- void ZapiszNowyPoJednym (int elem)
- int * ZwiekszOJeden ()
- int * Zwieksz2Razy ()

Public Attributes

- int * tab

Protected Attributes

- int n = 10
- int rozm = 0

4.12.1 Detailed Description

Definition at line 7 of file Tab.hh.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 Tab::Tab ( ) [inline]

Definition at line 15 of file Tab.hh.

4.12.2.2 Tab::~Tab ( ) [inline]

Definition at line 16 of file Tab.hh.

4.12.3 Member Function Documentation

4.12.3.1 int* & Tab::Poczatek ( ) [inline]

Definition at line 19 of file Tab.hh.

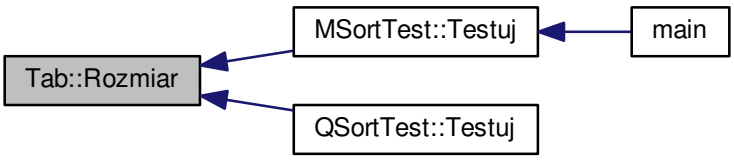
4.12.3.2 int& Tab::Pojemnosc ( ) [inline]

Definition at line 17 of file Tab.hh.

4.12.3.3 int& Tab::Rozmiar ( ) [inline]

Definition at line 18 of file Tab.hh.

Here is the caller graph for this function:



4.12.3.4 int * Tab::WypelnijOdNowa ( )

Definition at line 14 of file Tab.cpp.

4.12.3.5 void Tab::WypelnijTab ( ) [inline]

Definition at line 20 of file Tab.hh.

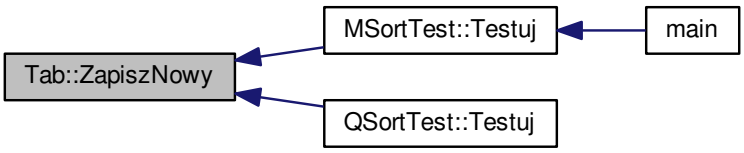
4.12.3.6 void Tab::WypiszTab ( ) [inline]

Definition at line 21 of file Tab.hh.

4.12.3.7 void Tab::ZapiszNowy (int *elem*)

Definition at line 39 of file Tab.cpp.

Here is the caller graph for this function:



4.12.3.8 void Tab::ZapiszNowyPoJednym (int *elem*)

Definition at line 47 of file Tab.cpp.

4.12.3.9 int * Tab::Zwieksz2Razy ( )

Definition at line 30 of file Tab.cpp.

4.12.3.10 int * Tab::ZwiekszOJeden ( )

Definition at line 4 of file Tab.cpp.

4.12.4 Member Data Documentation

4.12.4.1 int Tab::n = 10 [protected]

Definition at line 10 of file Tab.hh.

4.12.4.2 int Tab::rozm = 0 [protected]

Definition at line 11 of file Tab.hh.

4.12.4.3 int* Tab::tab

Definition at line 13 of file Tab.hh.

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

- Tab.hh
- Tab.cpp

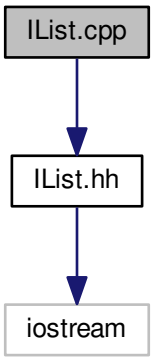
Chapter 5

File Documentation

5.1 IList.cpp File Reference

```
#include "IList.hh"
```

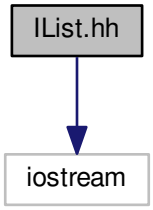
Include dependency graph for IList.cpp:



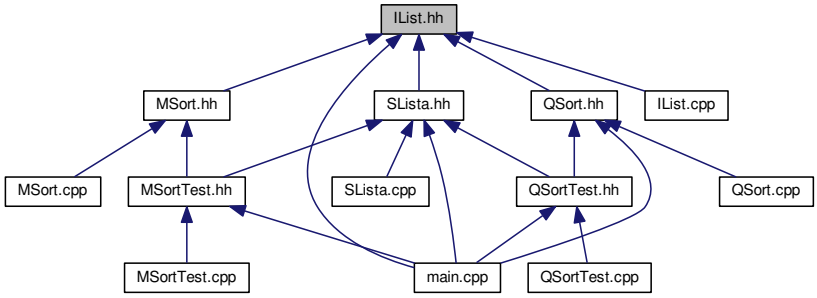
5.2 IList.hh File Reference

```
#include <iostream>
```

Include dependency graph for IList.hh:



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



Classes

- class IList< Object >

  Interfejs listy jednokierunkowej.

5.2.1 Detailed Description

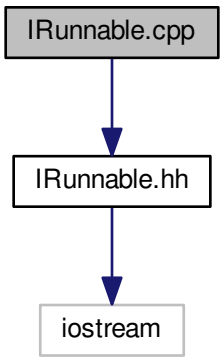
Plik zawiera interfejs listy jednokierunkową

Definition in file IList.hh.

5.3 IRunnable.cpp File Reference

```
#include "IRunnable.hh"
```

Include dependency graph for IRunnable.cpp:

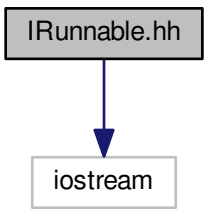


5.4 IRunnable.hh File Reference

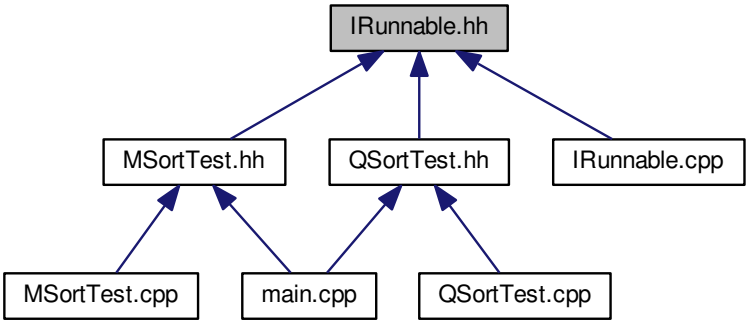
Interface do testowania algorytmow sortowania.

```
#include <iostream>
```

Include dependency graph for IRunnable.hh:



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

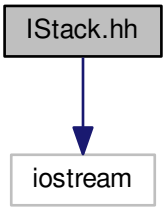


Classes

- class IRunnable

5.5 IStack.hh File Reference

`#include <iostream>`
Include dependency graph for IStack.hh:



Classes

- class IStack< Object >

  Klasa szablonowa modelująca interfejs stosu.

5.5.1 Detailed Description

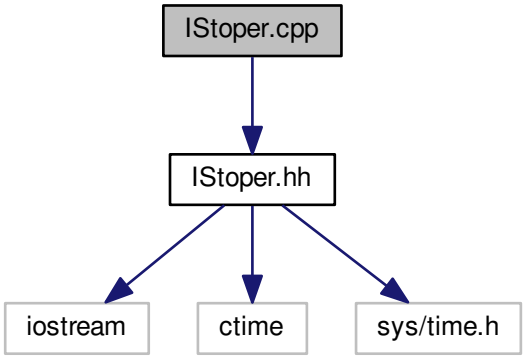
Plik zawiera interfejs stosu

Definition in file IStack.hh.

5.6 IStoper.cpp File Reference

```
#include "IStoper.hh"
```

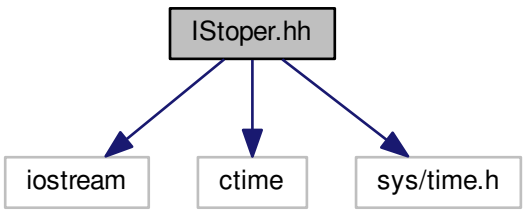
Include dependency graph for IStoper.cpp:



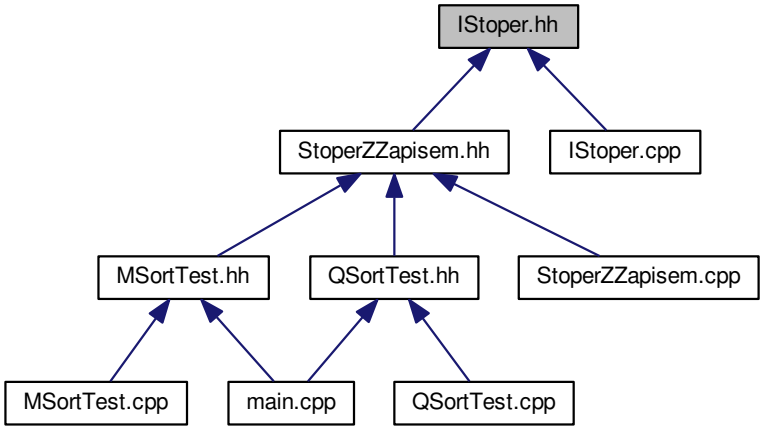
5.7 IStoper.hh File Reference

```
#include <iostream>
#include <ctime>
#include <sys/time.h>
```

Include dependency graph for IStoper.hh:



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



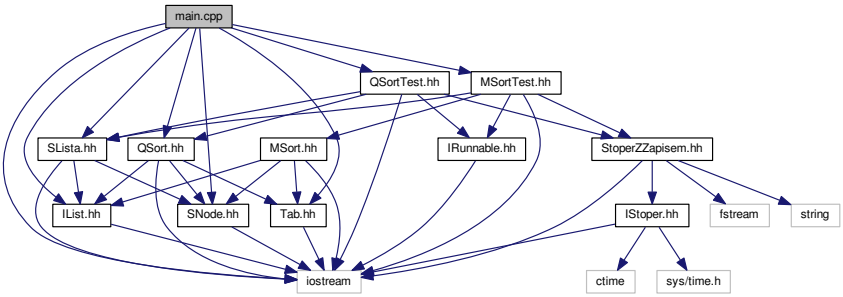
Classes

- class IStoper

5.8 main.cpp File Reference

```
#include <iostream>
#include "IList.hh"
#include "SLista.hh"
#include "SNode.hh"
#include "QSort.hh"
#include "Tab.hh"
#include "QSortTest.hh"
#include "MSortTest.hh"
```

Include dependency graph for main.cpp:



Functions

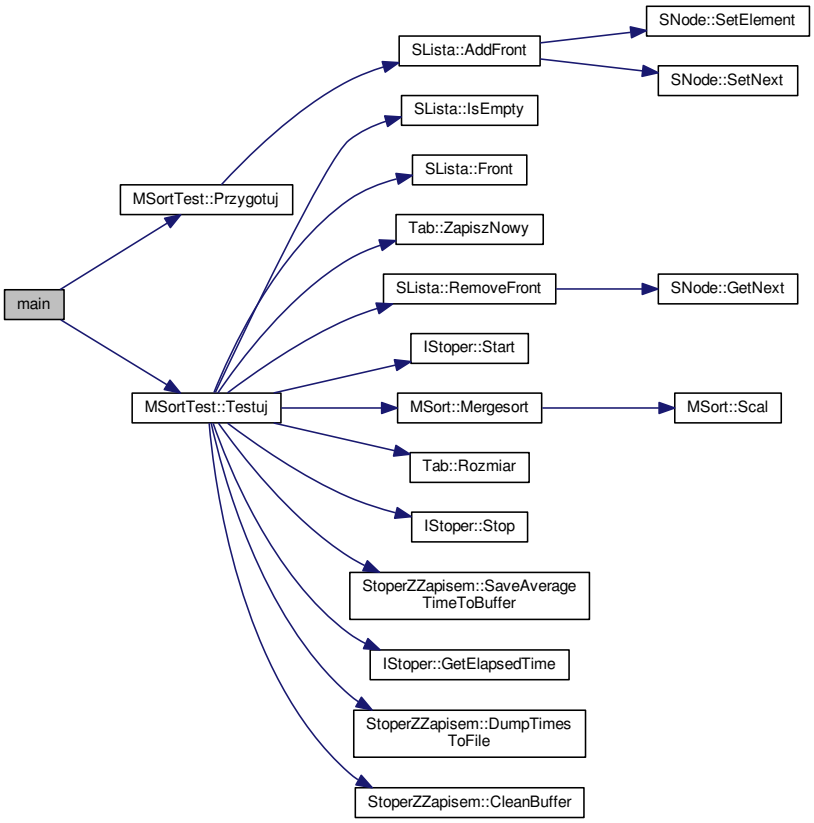
- int main ()

5.8.1 Function Documentation

5.8.1.1 int main ( )

Definition at line 17 of file main.cpp.

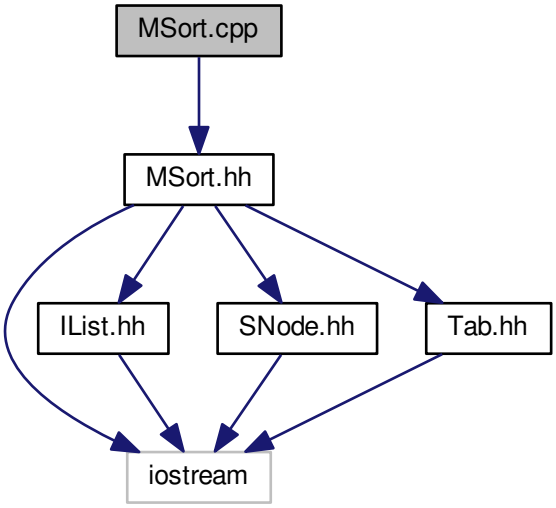
Here is the call graph for this function:



5.9 MSort.cpp File Reference

```
#include "MSort.hh"
```

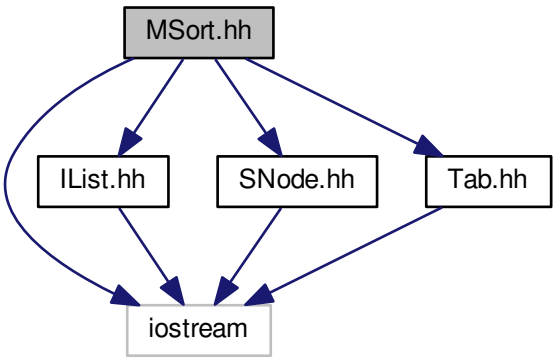
Include dependency graph for MSort.cpp:



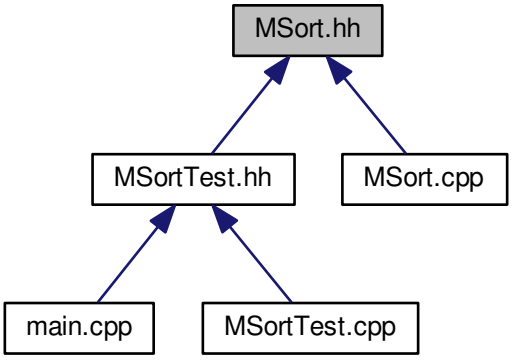
5.10 MSort.hh File Reference

```
#include <iostream>
#include "IList.hh"
#include "SNode.hh"
#include "Tab.hh"
```

Include dependency graph for MSort.hh:



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



Classes

- class MSort

  Merge Sort.

5.10.1 Detailed Description

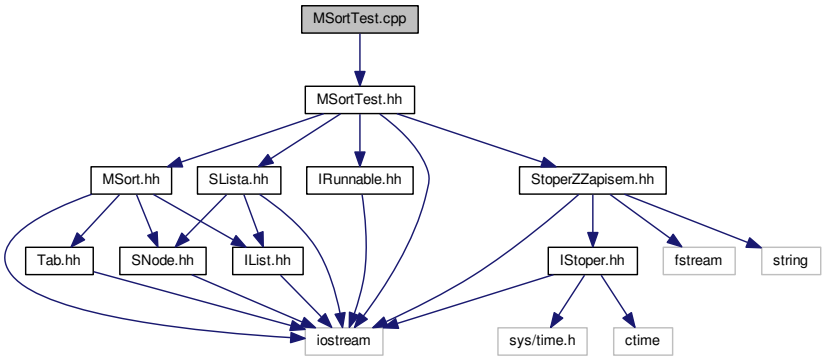
Plik zawiera definicję klasy implementującej algorytm sortowania przez scalanie

Definition in file MSort.hh.

5.11 MSortTest.cpp File Reference

```
#include "MSortTest.hh"
```

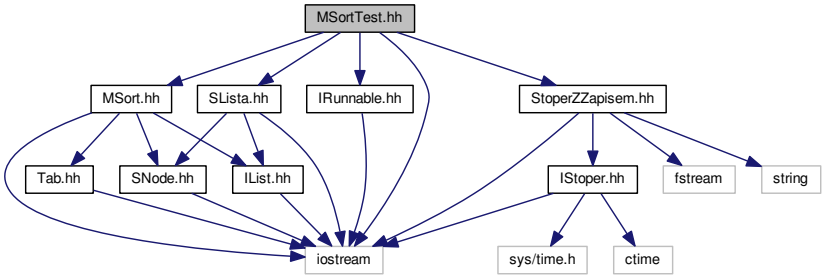
Include dependency graph for MSortTest.cpp:



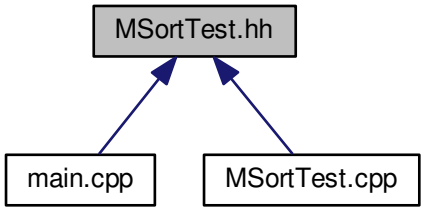
5.12 MSortTest.hh File Reference

```
#include <iostream>
#include "MSort.hh"
#include "SLista.hh"
#include "StoperZZapisem.hh"
#include "IRunnable.hh"
```

Include dependency graph for MSortTest.hh:



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



Classes

- class MSortTest

5.12.1 Detailed Description

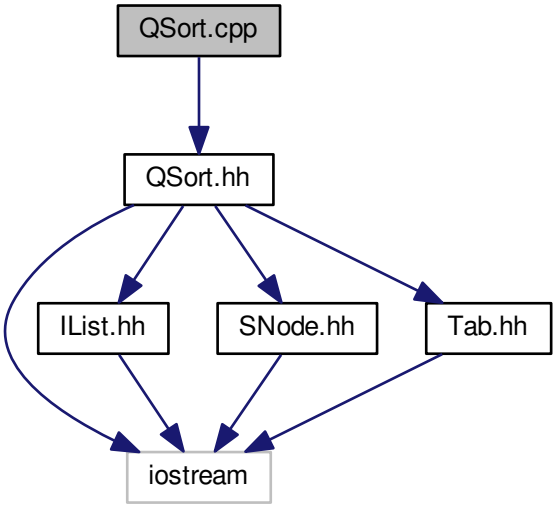
Klasa implementujaca testowanie algorytmu szybkiego sortowania

Definition in file MSortTest.hh.

5.13 QSort.cpp File Reference

```
#include "QSort.hh"
```

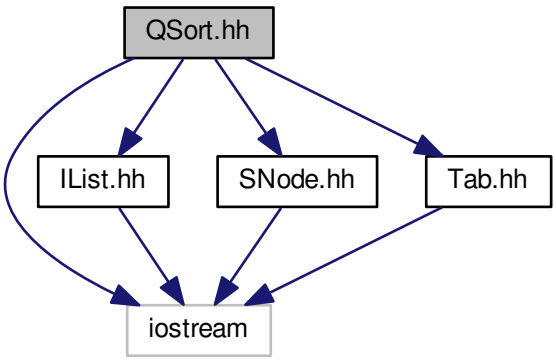
Include dependency graph for QSort.cpp:



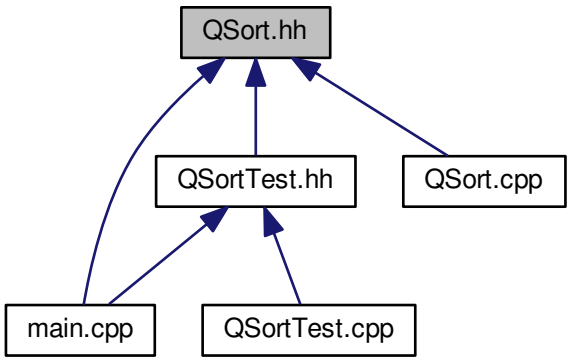
5.14 QSort.hh File Reference

```
#include <iostream>
#include "IList.hh"
#include "SNode.hh"
#include "Tab.hh"
```

Include dependency graph for QSort.hh:



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



Classes

- class QSort

  QuickSort.

5.14.1 Detailed Description

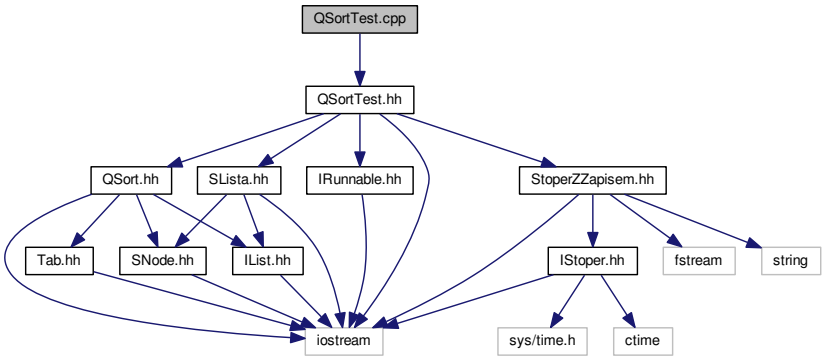
Plik zawiera definicję klasy implementującej algorytm szybkiego sortowania

Definition in file QSort.hh.

5.15 QSortTest.cpp File Reference

```
#include "QSortTest.hh"
```

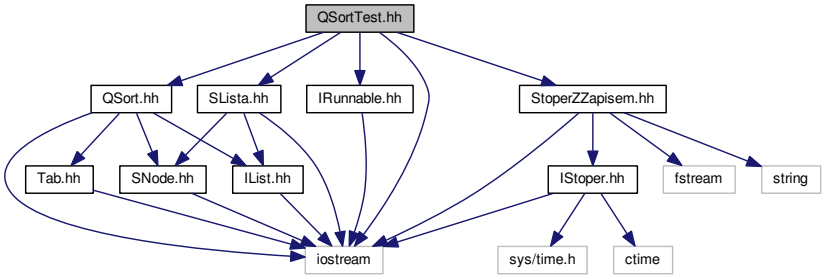
Include dependency graph for QSortTest.cpp:



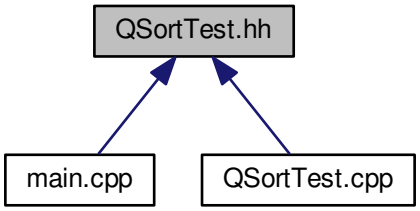
5.16 QSortTest.hh File Reference

```
#include <iostream>
#include "QSort.hh"
#include "SLista.hh"
#include "StoperZZapisem.hh"
#include "IRunnable.hh"
```

Include dependency graph for QSortTest.hh:



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



Classes

- class QSortTest

5.16.1 Detailed Description

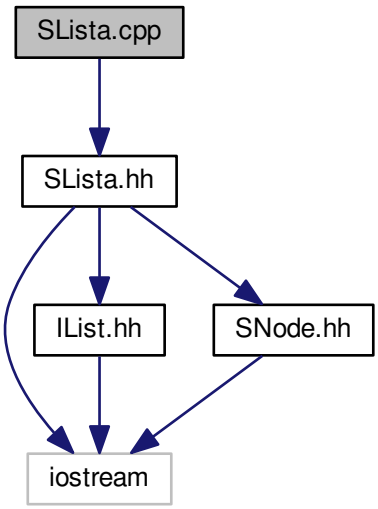
Klasa implementujaca testowanie algorytmu szybkiego sortowania

Definition in file QSortTest.hh.

5.17 SLista.cpp File Reference

```
#include "SLista.hh"
```

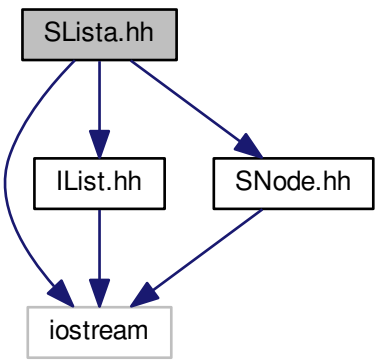
Include dependency graph for SLista.cpp:



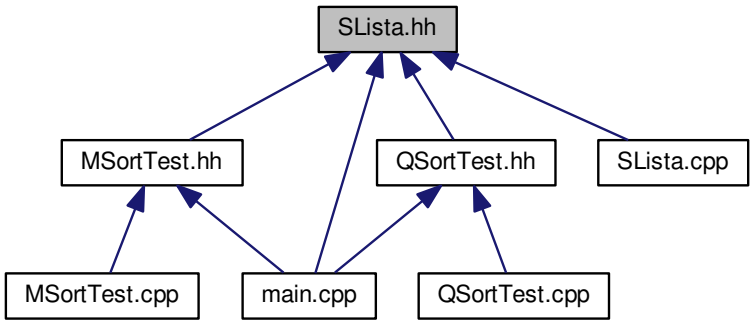
5.18 SLista.hh File Reference

```
#include <iostream>
#include "IList.hh"
#include "SNode.hh"
```

Include dependency graph for SLista.hh:



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



Classes

- class SLista< Object >

  Szablonowa klasa implementująca listę jednokierunkową

5.18.1 Detailed Description

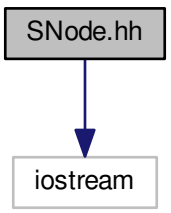
Plik zawiera definicję klasy implementującej listę jednokierunkową.

Definition in file SLista.hh.

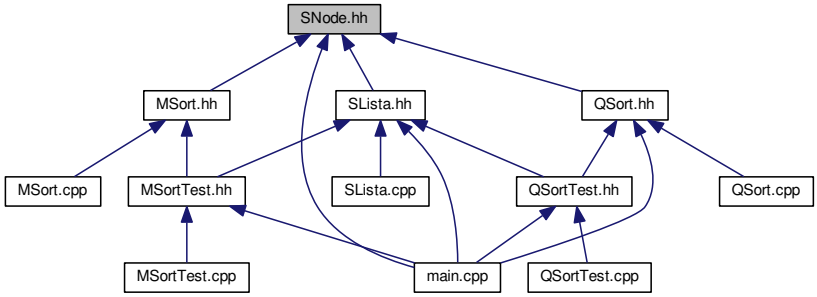
5.19 SNode.hh File Reference

```
#include <iostream>
```

Include dependency graph for SNode.hh:



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



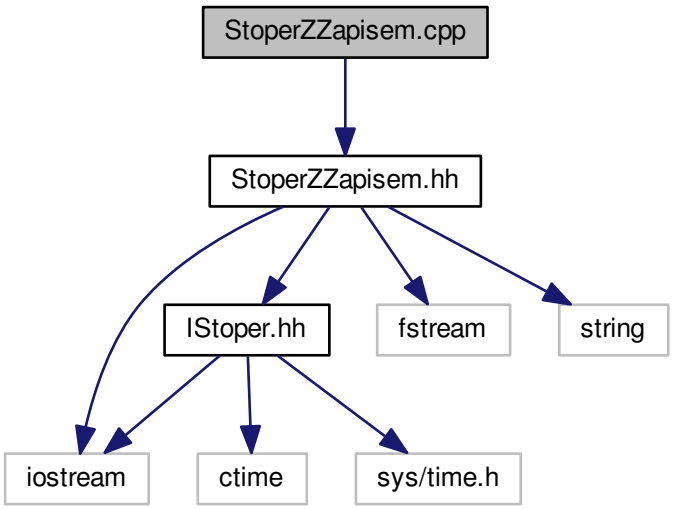
Classes

- class SNode< Object >

5.20 StoperZZapisem.cpp File Reference

```
#include "StoperZZapisem.hh"
```

Include dependency graph for StoperZZapisem.cpp:

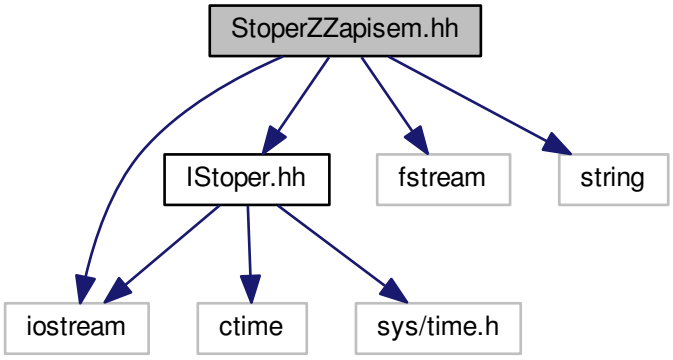


5.21 StoperZZapisem.hh File Reference

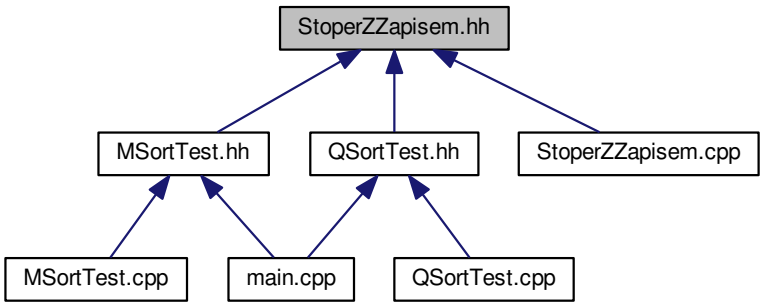
```
#include "IStoper.hh"
```

```
#include <iostream>
#include <fstream>
#include <string>
```

Include dependency graph for StoperZZapisem.hh:



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



Classes

- class StoperZZapisem

  Klasa implementująca rozbudowany stoper.

Macros

- #define POJEMNOSC 35
- #define BUFOR 6

5.21.1 Detailed Description

Plik zawiera implementację rozbudowanego stopera.

Definition in file StoperZZapisem.hh.

5.21.2 Macro Definition Documentation

5.21.2.1 #define BUFOR 6

Definition at line 11 of file StoperZZapisem.hh.

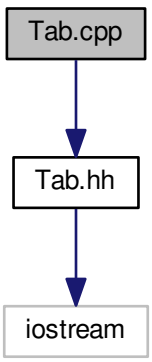
5.21.2.2 #define POJEMNOSC 35

Definition at line 10 of file StoperZZapisem.hh.

5.22 Tab.cpp File Reference

```
#include "Tab.hh"
```

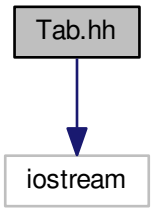
Include dependency graph for Tab.cpp:



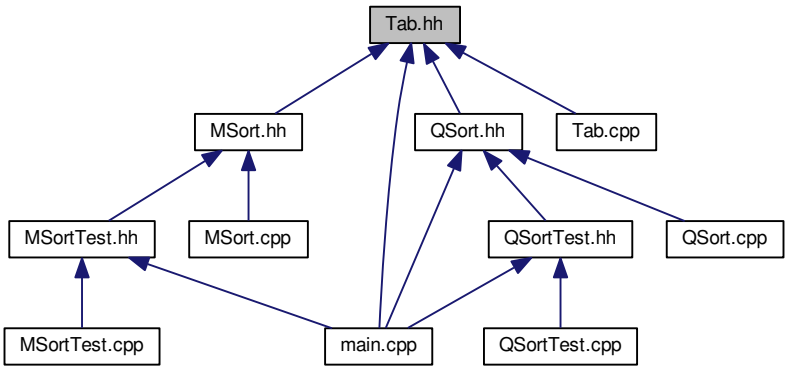
5.23 Tab.hh File Reference

```
#include <iostream>
```

Include dependency graph for Tab.hh:



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



Classes

- class Tab