Mnozenie_tablic

Wygenerowano przez Doxygen 1.7.6.1

Fri Mar 21 2014 12:19:23

Spis treści

Rozdział 1

Mnożenie tablic

Autor

Krzysztof Kucharczyk

Data

11.03.2014

Wersja

3

Program umożlwia operacje na specjalnych obiektach Obiekt, które składają się z wskaźnika i kilku przydatnych metod.

Rozdział 2

Indeks klas

2.1 Lista klas

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

Rozdział 3

Indeks plików

3.1 Lista plików

Tutaj znajduje się lista wszystkich udokumentowanych plików z ich krótkimi opisami:

Rozdział 4

Dokumentacja klas

4.1 Dokumentacja klasy Obiekt

Klasa umożliwia tworzenie przydatnej struktury.

`#include <obiekt.h>`

Metody publiczne

- Obiekt ()

  Inicjuje obiekt pustym wskaźnikiem.
- Obiekt (int ∗Wskaznik)

  Inicjuje obiekt określonym wskaźnikiem.
- Obiekt operator+ (const Obiekt &Nowy)

  Pozwala dodawać dwa obiekty.
- Obiekt & operator= (const Obiekt &Nowy)

  Pozwala przypisywać jeden obiekt do drugiego.
- bool operator== (const Obiekt &Nowy)

  Pozwala intuicyjnie porównywać obiekty.
- void Show ()

  Umożlwia zwizualizowanie elementów obiektu.
- void Pobierz_dane (string nazwa_pliku)

  Pobiera dane z określonego pliku.
- void Pomnoz (int mnoznik)

  Mnoży elemnty tablicy Obiektu.
- void Czy_rowne (const Obiekt &Porownywany)

  Sprawdza, czy Obiekty są sobie równe.
- void Zamien_elementy (int i, int j)

  Zamienia elementy o podanych indeksach.
- void Odwroc_kolejnosc ()

Odwraca kolejność tablicy.

- void Dodaj_element (int element)

  Dodaje jeden element do tablicy.

- void Dodaj_elementy (Obiekt &Nowy)

  Dodaje wiele elementów do tablicy.

Atrybuty publiczne

- int * Tablica

  Umożliwia tworzenie uniwersalnych tablicy.

4.1.1 Opis szczegółowy

Klasa umożliwia tworzenie przydatnej struktury.

Klasa pozwala na tworzenie bardzo przydatnej, uniwersalnej struktury bardzo podatnej na wszelki modyfikacje (np. sortowanie). Jej najważniejszym elementem jest wskaźnik umozliwiający stworzenie dynamicznej tablicy. Dane w niej przechowywane mogą być w różnoraki sposób modyfikowane dzięki zaimplementowanym metodom.

4.1.2 Dokumentacja konstruktora i destruktora

4.1.2.1 Obiekt::Obiekt ( ) [inline]

Inicjuje obiekt pustym wskaźnikiem.

Prosty konstruktor, jego zadanie ogranicza się do dezaktywowania wskaźnika, by w razie czego na nic nie pokazywał i nie wprowadzał w ewentualny błąd.

4.1.2.2 Obiekt::Obiekt (int * *Wskaznik*) [inline]

Inicjuje obiekt określonym wskaźnikiem.

Kontruktor pozwala zainicjować obiekt już istniejącą tablicą danych, dzięki czemu nie trzeba jej ponownie przepisywać.

4.1.3 Dokumentacja funkcji składowych

4.1.3.1 void Obiekt::Czy_rowne (const Obiekt & *Porownywany*)

Sprawdza, czy Obiekty są sobie równe.

Metoda sprawdza, czy dwa obiekty są sobie równe. Milczy w przypaku optymistycznym, wyświetla stosowny komunikat w przypadku przeciwnym.

Parametry

in	- *Porownywany*	Nazwa porównywanego obiektu

4.1.3.2 void Obiekt::Dodaj_element (int *element*)

Dodaje jeden element do tablicy.

Metoda dodaje jeden element na koniec tablicy poprzez stworzenie nowej, o jeden większej tablicy, do której przepisywane są dane ze starej i dodawany nowy element na samym końcu. Stara tablica zostaje usunięta. Oczywiście element zerowy zostaje adekwatnie uaktualniony.

Parametry

in	*element*	Nowy element, który zostanie dodany na koniec tablicy

4.1.3.3 void Obiekt::Dodaj_elementy (Obiekt & *Nowy*)

Dodaje wiele elementów do tablicy.

Metoda umożliwia dodanie dużej ilości elementów na koniec istniejącej w obiekcie tablicy. Działa na zasadzie stworzenia nowej, większej tablicy i umieszczeniu w niej elementów obu tablic. Stara tablica zostaje zwolniona, a indeks zerowy uaktualniony.

Parametry

Nowy	Obiekt Obiekt, którego tablica zostaje wpisana na koniec istniejącej tablicy w Obiekcie.

4.1.3.4 void Obiekt::Odwroc_kolejnosc ( )

Odwraca kolejność tablicy.

Metoda powoduje inwersję elementów tablicy. Oczywiście element zerowy zostaje na miejscu.

4.1.3.5 Obiekt Obiekt::operator+ (const Obiekt & *Nowy*)

Pozwala dodawać dwa obiekty.

Przeciążenie to pozwala dodawać tablicę jednego obiektu do końca tablicy drugiego obiektu. Dzięki modyfikatorowi const obiekt dodawany ma zapewnioną nietykalność.

4.1.3.6 Obiekt & Obiekt::operator= (const Obiekt & *Nowy*)

Pozwala przypisywać jeden obiekt do drugiego.

Przeciążenie umożliwia w prosty sposób przypisanie wartości jednego obiektu do drugiego. Obiekt, który jest przepisywany jest nietykalny dzięki modyfikatorowi const.

4.1.3.7 bool Obiekt::operator== (const Obiekt & *Nowy*)

Pozwala intuicyjnie porównywać obiekty.

Przeciążenie umożliwia intuicyjne porównywanie dwóch elementów. Zwracana jest jedna z dwóch wartości:

- 1 - gdy oba obiekty są identyczne,
- 0 - gdy oba obiekty są różne.

4.1.3.8 void Obiekt::Pobierz_dane (string *nazwa_pliku*)

Pobiera dane z określonego pliku.

Metoda umożliwia pobranie danych pliku o określonej nazwie. Plik jest sprawdzany pod względem swojego formatu, tj. czy zachowany jest następujący układ:

```
* indeks
* 1.              | ilość elementów
* 2.              | element_1
* 3.             | element_2
* .
* .
* .
* ilość_elementów+1 | element ilość_elementów-1
*
```

Parametry

in	*nazwa_pliku*	Zawiera nazwę pliku, z którego pobierane są dane

4.1.3.9 void Obiekt::Pomnoz (int *mnoznik*)

Mnoży elemnty tablicy Obiektu.

Metoda służy do multiplikowania elementów tablicy przez określoną liczbę.

Parametry

in	*mnoznik*	Wartość, o jaką mają zostać pomnożone elementy

4.1.3.10 void Obiekt::Show ( )

Umożlwia zwizualizowanie elementów obiektu.

Metoda pozwala wypisać całą zawartość obiektu (z wyjątkiem elementu zerowego, tj. długośvi tablicy).

4.1.3.11 void Obiekt::Zamien_elementy (int *i*, int *j*)

Zamienia elementy o podanych indeksach.

Metoda zamienia miejscami elementy tablicy o podanych indeksach. Jedynym ograniczeniem jest indeks zerowy, który zawiera informację o długości tablicy. Jego zmiana jest niemożliwa, o tym także poinformuje program w razie ewentualnego błędu.

Parametry

in	*i*	Indeks pierwszego elementu
in	*j*	Indeks drugiego elementu

4.1.4 Dokumentacja atrybutów składowych

4.1.4.1 int* Obiekt::Tablica

Umożliwia tworzenie uniwersalnych tablicy.

Pole pozwala na tworzenie uniwersalnych tablic o zdefiniowanych przez użytkownika wielkościach. Pole to jest podstawowym polem klasy.

Dokumentacja dla tej klasy została wygenerowana z plików:

- /home/krzysztof/Desktop/PAMSI/Laboratorium/Laboratorium_2/Mnozenie_tablic/inc/obiekt.h
- /home/krzysztof/Desktop/PAMSI/Laboratorium/Laboratorium_2/Mnozenie_tablic/src/obiekt.cpp

Rozdział 5

Dokumentacja plików

5.1 Dokumentacja pliku /home/krzysztof/Desktop/PAMSI/Laboratorium/-Laboratorium_2/Mnozenie_tablic/inc/obiekt.h

Zawiera definicję klasy Obiekt.

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

Komponenty

- class Obiekt

  Klasa umożliwia tworzenie przydatnej struktury.

5.1.1 Opis szczegółowy

Zawiera definicję klasy Obiekt. Plik zawiera definicję klasy Obiekt.