

ProgramowanieZaawansowane4

Generated by Doxygen 1.15.0



---

<b>1 Class Index</b>	<b>1</b>
1.1 Class List	1
<b>2 File Index</b>	<b>3</b>
2.1 File List	3
<b>3 Class Documentation</b>	<b>5</b>
3.1 matrix Class Reference	5
3.1.1 Detailed Description	6
3.1.2 Constructor & Destructor Documentation	6
3.1.2.1 matrix() [1/3]	6
3.1.2.2 matrix() [2/3]	7
3.1.2.3 matrix() [3/3]	7
3.1.3 Member Function Documentation	7
3.1.3.1 alokuj()	7
3.1.3.2 operator=()	8
3.1.3.3 pokaz()	9
3.1.3.4 size()	9
3.1.3.5 wstaw()	9
<b>4 File Documentation</b>	<b>11</b>
4.1 file_utils.h	11
4.2 matrix.h	11
<b>Index</b>	<b>13</b>



# Chapter 1

## Class Index

### 1.1 Class List

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

<a href="#">matrix</a>	Klasa reprezentująca macierz kwadratowa . . . . .	<a href="#">5</a>
------------------------	---	-------------------



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all documented files with brief descriptions:

include/ <a href="#">file_utils.h</a> . . . . .	11
include/ <a href="#">matrix.h</a> . . . . .	11





## Chapter 3

# Class Documentation

### 3.1 matrix Class Reference

Klasa reprezentująca macierz kwadratowa.

```
#include <matrix.h>
```

#### Public Member Functions

- **matrix** ()  
*Konstruktor domyslny. Tworzy pusta macierz o rozmiarze 0.*
- **matrix** (int n)  
*Konstruktor alokujacy pamiec. Tworzy macierz o wymiarach  $n \times n$  wypelniona zerami.*
- **matrix** (int n, int \*t)  
*Konstruktor inicjalizujacy macierz danymi z tablicy.*
- **matrix** (const **matrix** &m)  
*Konstruktor kopiujacy (Deep Copy). Tworzy nowy obiekt i kopiuje do niego zawartosc innej macierzy.*
- **~matrix** ()  
*Destruktor. Pamiec zwalniana jest automatycznie przez unique\_ptr.*
- **matrix** & **alokuj** (int n)  
*Alokuje pamiec dla macierzy.*
- **matrix** & **wstaw** (int x, int y, int wartosc)  
*Wstawia wartosc do macierzy.*
- int **pokaz** (int x, int y) const  
*Pobiera wartosc z macierzy.*
- int **size** () const  
*Zwraca aktualny rozmiar macierzy.*
- **matrix** & **operator=** (const **matrix** &m)  
*Operator przypisania kopiujacego (Deep Copy). Obsluguje operacje:  $A = B$ .*
- **matrix** & **losuj** ()
- **matrix** & **losuj** (int ilosc)
- **matrix** & **dowroc** ()
- **matrix** & **przekatna** ()
- **matrix** & **pod\_przekatna** ()
- **matrix** & **nad\_przekatna** ()
- **matrix** & **szachownica** ()

- `matrix & diagonalna` (int \*t)
- `matrix & diagonalna_k` (int k, int \*t)
- `matrix & kolumna` (int x, int \*t)
- `matrix & wiersz` (int y, int \*t)
- `matrix operator+` (const `matrix` &m) const
- `matrix operator-` (const `matrix` &m) const
- `matrix operator*` (const `matrix` &m) const
- `matrix operator+` (int a) const
- `matrix operator-` (int a) const
- `matrix operator*` (int a) const
- `matrix & operator++` (int)
- `matrix & operator--` (int)
- `matrix & operator+=` (int a)
- `matrix & operator-=` (int a)
- `matrix & operator*=` (int a)
- `matrix & operator()` (double a)

*Funktor: Powieksza wszystkie elementy o czesc calkowita z double.*

- `bool operator==` (const `matrix` &m) const
- `bool operator>` (const `matrix` &m) const
- `bool operator<` (const `matrix` &m) const

## Friends

- `matrix operator+` (int a, const `matrix` &m)
- `matrix operator-` (int a, const `matrix` &m)
- `matrix operator*` (int a, const `matrix` &m)
- `std::ostream & operator<<` (std::ostream &o, const `matrix` &m)

*Operator strumieniowy do wypisywania macierzy.*

## 3.1.1 Detailed Description

Klasa reprezentująca macierz kwadratowa.

- Klasa zarządza pamięcią dynamiczną przy użyciu inteligentnego wskaźnika `std::unique_ptr`. Przechowuje dane w tablicy jednowymiarowej.

## 3.1.2 Constructor & Destructor Documentation

### 3.1.2.1 `matrix()` [1/3]

```
matrix::matrix (
    int n) [explicit]
```

Konstruktor alokujący pamięć. Tworzy macierz o wymiarach  $n \times n$  wypełnioną zerami.

#### Parameters

---

$n$	Zadana wielkosc macierzy.
-----	---------------------------

### 3.1.2.2 matrix() [2/3]

```
matrix::matrix (  
    int n,  
    int * t)
```

Konstruktor inicjalizujacy macierz danymi z tablicy.

#### Parameters

$n$	Wielkosc macierzy.
$t$	Wskaznik do tablicy z danymi.

### 3.1.2.3 matrix() [3/3]

```
matrix::matrix (  
    const matrix & m)
```

Konstruktor kopiujacy (Deep Copy). Tworzy nowy obiekt i kopiuje do niego zawartosc innej macierzy.

#### Parameters

$m$	Macierz zrodlowa.
-----	-------------------

## 3.1.3 Member Function Documentation

### 3.1.3.1 alokuj()

```
matrix & matrix::alokuj (  
    int n)
```

Alokuje pamiec dla macierzy.

- Jesli macierz nie ma pamieci -> alokuje. Jesli ma inna wielkosc -> realokuje. Jesli ma te sama wielkosc -> nie robi nic (optymalizacja).

•

#### Parameters

$n$	Nowy wymiar macierzy.
-----	-----------------------

#### Returns

matrix& Referencja do biezacego obiektu.

### 3.1.3.2 operator=()

```
matrix & matrix::operator= (  
    const matrix & m)
```

Operator przypisania kopiujacego (Deep Copy). Obsluguje operacje:  $A = B$ .

#### Parameters

---

<i>m</i>	Macierz zrodlowa.
----------	-------------------

**Returns**

matrix& Referencja do this.

**3.1.3.3 pokaz()**

```
int matrix::pokaz (  
    int x,  
    int y) const
```

Pobiera wartosc z macierzy.

- Parameters**

<i>x</i>	Numer kolumny.
<i>y</i>	Numer wiersza.

**Returns**

int Wartosc komorki.

**3.1.3.4 size()**

```
int matrix::size () const
```

Zwraca aktualny rozmiar macierzy.

**Returns**

int Wymiar n.

**3.1.3.5 wstaw()**

```
matrix & matrix::wstaw (  
    int x,  
    int y,  
    int wartosc)
```

Wstawia wartosc do macierzy.

- Parameters**

<i>x</i>	Numer kolumny.
<i>y</i>	Numer wiersza.
<i>wartosc</i>	Liczba do wstawienia.

**Returns**

matrix& Referencja do biezacego obiektu (umożliwia lancuchowanie).

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

- include/matrix.h
- src/matrix\_core.cpp
- src/matrix\_logic.cpp
- src/matrix\_math.cpp

# Chapter 4

## File Documentation

### 4.1 file\_utils.h

```
00001 #pragma once
00002 #include <vector>
00003 #include <string>
00004
00005 // Forward declaration (Zapowiedz klasy)
00006 // Dzięki temu nie musimy robic #include "matrix.h" w tym pliku,
00007 // co przyspiesza kompilacje i zapobiega cyklom zalaczen.
00008 class matrix;
00009
00015 std::vector<int> wczytajDaneZPliku(const std::string& sciezka);
00016
00022 void zapiszMacierzDoPliku(const std::string& sciezka, const matrix& m);
```

### 4.2 matrix.h

```
00001 #pragma once
00002 #include <memory>
00003 #include <iostream>
00004
00010 class matrix {
00011 private:
00012     int n;
00013     std::unique_ptr<int[]> data;
00014
00023     int& at(int x, int y);
00024
00031     int at(int x, int y) const;
00032
00033 public:
00034     // --- Konstruktory i Destruktor ---
00035
00040     matrix();
00041
00047     explicit matrix(int n);
00048
00054     matrix(int n, int* t);
00055
00061     matrix(const matrix& m);
00062
00067     ~matrix();
00068
00069     // --- Zarzadzanie Pamiecia i Dostep ---
00070
00079     matrix& alokuj(int n);
00080
00088     matrix& wstaw(int x, int y, int wartosc);
00089
00096     int pokaz(int x, int y) const;
00097
00102     int size() const;
00103
00104     // --- Operatory Przypisania ---
00105
```

```

00112     matrix& operator=(const matrix& m);
00113
00114     // --- Metody Logiczne i Wzorce ---
00115
00116     matrix& losuj();
00117     matrix& losuj(int ilosc);
00118     matrix& dowroc(); // Transpozycja
00119
00120     // Wzorce 0/1
00121     matrix& przekatna();
00122     matrix& pod_przekatna();
00123     matrix& nad_przekatna();
00124     matrix& szachownica();
00125
00126     // Wzorce z tablicy zewnetrznej
00127     matrix& diagonalna(int* t);
00128     matrix& diagonalna_k(int k, int* t);
00129     matrix& kolumna(int x, int* t);
00130     matrix& wiersz(int y, int* t);
00131
00132     // --- Operatory Arytmetyczne (Macierz <-> Macierz) ---
00133     // Zwracamy nowy obiekt przez wartosc
00134
00135     matrix operator+(const matrix& m) const;
00136     matrix operator-(const matrix& m) const;
00137     matrix operator*(const matrix& m) const;
00138
00139     // --- Operatory Arytmetyczne (Macierz <-> Skalar) ---
00140
00141     matrix operator+(int a) const;
00142     matrix operator-(int a) const;
00143     matrix operator*(int a) const;
00144
00145     // --- Operatory Modyfikujace (zwracaja *this) ---
00146
00147     matrix& operator++(int); // Postfiks A++
00148     matrix& operator--(int); // Postfiks A--
00149     matrix& operator+=(int a);
00150     matrix& operator-=(int a);
00151     matrix& operator*=(int a);
00152
00153     matrix& operator()(double a);
00154
00155     // --- Operatory Porownania ---
00156
00157     bool operator==(const matrix& m) const;
00158     bool operator>(const matrix& m) const;
00159     bool operator<(const matrix& m) const;
00160
00161     // --- Funkcje Zaprzyjaznione (Friends) ---
00162     // Pozwalaja na operacje typu: 5 + A (gdzie int jest po lewej stronie)
00163
00164     friend matrix operator+(int a, const matrix& m);
00165     friend matrix operator-(int a, const matrix& m);
00166     friend matrix operator*(int a, const matrix& m);
00167
00168     friend std::ostream& operator<<(std::ostream& o, const matrix& m);
00169
00170 };

```



# Index

- alokuj
  - matrix, [7](#)
- include/file\_utils.h, [11](#)
- include/matrix.h, [11](#)
- matrix, [5](#)
  - alokuj, [7](#)
  - matrix, [6](#), [7](#)
  - operator=, [7](#)
  - pokaz, [9](#)
  - size, [9](#)
  - wstaw, [9](#)
- operator=
  - matrix, [7](#)
- pokaz
  - matrix, [9](#)
- size
  - matrix, [9](#)
- wstaw
  - matrix, [9](#)