

Matrix Calculator

Generated by Doxygen 1.9.3

1 File Index	1
1.1 File List	1
2 File Documentation	3
2.1 main.cpp File Reference	3
2.1.1 Enumeration Type Documentation	4
2.1.1.1 operation	4
2.1.2 Function Documentation	4
2.1.2.1 Add()	4
2.1.2.2 Cofactor()	5
2.1.2.3 convert()	5
2.1.2.4 debugMatrix()	5
2.1.2.5 Determinant()	6
2.1.2.6 Inverse()	6
2.1.2.7 main()	6
2.1.2.8 Multiply()	7
2.1.2.9 read_matrix()	7
2.1.2.10 save_solution() [1/3]	7
2.1.2.11 save_solution() [2/3]	8
2.1.2.12 save_solution() [3/3]	8
2.1.2.13 Subtract()	8
2.1.2.14 Transpose()	9
2.2 Matrix.h File Reference	9
2.2.1 Function Documentation	9
2.2.1.1 Add()	9
2.2.1.2 Cofactor()	10
2.2.1.3 debugMatrix()	10
2.2.1.4 Determinant()	10
2.2.1.5 Inverse()	11
2.2.1.6 Multiply()	11
2.2.1.7 read_matrix()	11
2.2.1.8 save_solution() [1/3]	12
2.2.1.9 save_solution() [2/3]	12
2.2.1.10 save_solution() [3/3]	12
2.2.1.11 Subtract()	13
2.2.1.12 Transpose()	13
2.3 Matrix.h	13
Index	15

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

main.cpp	3
Matrix.h	9

Chapter 2

File Documentation

2.1 main.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <cmath>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "matrix.h"
```

Enumerations

- enum class [operation](#) {
 [add](#) = 0 , [subtract](#) , [multiply](#) , [determinant](#) ,
 [inverse](#) , [cofactor](#) , [transpose](#) , [random](#) }

Functions

- void [debugMatrix](#) (const std::vector< std::vector< int > > vect)
- const std::vector< std::vector< int > > [read_matrix](#) (std::string file_name)
- void [save_solution](#) (const std::vector< std::vector< int > > solution, std::string &outputFile)
- void [save_solution](#) (int solution, std::string &outputFile)
- void [save_solution](#) (std::string input, std::string &outputFile)
- int [Determinant](#) (std::vector< std::vector< int > > A)
- void [Add](#) (const std::vector< std::vector< int > > A, const std::vector< std::vector< int > > B, std::string &outputFile)
- void [Subtract](#) (const std::vector< std::vector< int > > A, const std::vector< std::vector< int > > B, std::string &outputFile)
- void [Multiply](#) (const std::vector< std::vector< int > > A, const std::vector< std::vector< int > > B, std::string &outputFile)
- std::vector< std::vector< int > > [Transpose](#) (const std::vector< std::vector< int > > A)
- std::vector< std::vector< int > > [Cofactor](#) (const std::vector< std::vector< int > > A)
- void [Inverse](#) (const std::vector< std::vector< int > > A, std::string &outputFile)
- [operation convert](#) (std::string &perform)
- int [main](#) (int argc, char *argv[])

2.1.1 Enumeration Type Documentation

2.1.1.1 operation

```
enum class operation [strong]
```

An Enum class of operations

Parameters

<i>add</i>	
<i>subtract</i>	
<i>multiply</i>	
<i>determinant</i>	
<i>inverse</i>	
<i>cofactor</i>	
<i>transpose</i>	
<i>random</i>	

Enumerator

add	
subtract	
multiply	
determinant	
inverse	
cofactor	
transpose	
random	

2.1.2 Function Documentation

2.1.2.1 Add()

```
void Add (
    const std::vector< std::vector< int > > A,
    const std::vector< std::vector< int > > B,
    std::string & outputFile )
```

The function calculates the addition of two matrix and returns a solution to the file

Parameters

<i>const</i>	std::vector<std::vector<int>> A first matrix from file
<i>const</i>	std::vector<std::vector<int>> B second matrix from file

Returns

`std::vector<std::vector<int>>` solution solution from the add operation

2.1.2.2 Cofactor()

```
std::vector< std::vector< int > > Cofactor (
    const std::vector< std::vector< int > > A )
```

The function takes a matrix and returns the cofactor of the matrix

Parameters

<i>const</i>	<code>std::vector<std::vector<int>></code> matrix The input matrix to be transposed
--------------	-------------------------------------------------------------------------------------------------

Returns

`std::vector<std::vector<int>>` solution returns the cofactor matrix

2.1.2.3 convert()

```
operation convert (
    std::string & perform )
```

The function takes a string input and compares with default parameters to get the operation type

Parameters

<code>std::string&</code>	perform pointer with the string function
-------------------------------	------------------------------------------

Returns

operation convert

2.1.2.4 debugMatrix()

```
void debugMatrix (
    const std::vector< std::vector< int > > vect )
```

The function prints a matrix on console

Parameters

<i>const</i>	<code>std::vector<std::vector<double>></code> vect vector of vector double printed to the output
--------------	--------------------------------------------------------------------------------------------------------------

2.1.2.5 Determinant()

```
int Determinant (
    std::vector< std::vector< int > > A )
```

The function calculates the determinate of a matrix and returns a solution

Parameters

<code>std::vector<std::vector<double>></code>	A a vector of vector matrix containing the values read from file
-----------------------------------------------------------	------------------------------------------------------------------

Returns

`int Determinant` returns the value of the determinant

2.1.2.6 Inverse()

```
void Inverse (
    const std::vector< std::vector< int > > A,
    std::string & outputFile )
```

The function calculates the inverse of a matrix and returns a matrix of vector containing the solution

Parameters

<i>const</i>	<code>std::vector<std::vector<int>></code> matrix The input matrix
<i>std::string&</i>	outputFile pointer to the file where output solution from the operation is saved

2.1.2.7 main()

```
int main (
    int argc,
    char * argv[] )
```

2.1.2.8 Multiply()

```
void Multiply (
    const std::vector< std::vector< int > > A,
    const std::vector< std::vector< int > > B,
    std::string & outputFile )
```

The function multiplies two matrix and returns a matrix of vector containing the solution to the file

Parameters

<i>const</i>	std::vector<std::vector<int>> A first Vector of matrix to be multiplied
<i>const</i>	std::vector<std::vector<int>> B second vector of matrix to be multiplied
<i>std::string&</i>	outputFile pointer to the file where output solution from the operation is saved

2.1.2.9 read_matrix()

```
const std::vector< std::vector< int > > read_matrix (
    std::string file_name )
```

The function reads a matrix of random number from a file

Parameters

<i>const</i>	std::string& file_name pointer to the file been read from
--------------	-----------------------------------------------------------

Returns

std::vector< std::vector<double>> read_matrix returns a vector of vector matrix containing the read matrix

Extract numbers from text

2.1.2.10 save_solution() [1/3]

```
void save_solution (
    const std::vector< std::vector< int > > solution,
    std::string & outputFile )
```

The function saves the solution form a matrix operation to a file

Parameters

<i>const</i>	std::string& file_name pointer to the file been saved to
<i>const</i>	std::vector< std::vector<double>> solution vector of vector matrix containing the matrix to be saved to file

2.1.2.11 save_solution() [2/3]

```
void save_solution (
    int solution,
    std::string & outputFile )
```

The function saves the solution form a matrix operation to a file

Parameters

<i>double</i>	solution double with value to the printed to file
<i>std::string&</i>	outputFile pointer to the file been saved to

2.1.2.12 save_solution() [3/3]

```
void save_solution (
    std::string input,
    std::string & outputFile )
```

The function saves the solution form a matrix operation to a file

Parameters

<i>std::string</i>	file_name string with error message to the printed to file
<i>std::string</i>	&outputFile pointer to the file been saved to

2.1.2.13 Subtract()

```
void Subtract (
    const std::vector< std::vector< int > > A,
    const std::vector< std::vector< int > > B,
    std::string & outputFile )
```

The function calculates the difference of two matrix and returns a solution to the file

Parameters

<i>const</i>	std::vector<std::vector<int>> matrix1 first matrix from file
<i>const</i>	std::vector<std::vector<int>> matrix2 second matrix from file
<i>std::string&</i>	outputFile pointer to the file where output solution from the operation is saved

2.1.2.14 Transpose()

```
std::vector< std::vector< int > > Transpose (
    const std::vector< std::vector< int > > A )
```

The function takes a matrix and returns the matrix transpose to the file

Parameters

<i>const</i>	std::vector<std::vector<int>> matrix The input matrix to be transposed
--------------	------------------------------------------------------------------------

Returns

std::vector<std::vector<int>> solution returns the matrix transpose

2.2 Matrix.h File Reference

```
#include <vector>
```

Functions

- const std::vector< std::vector< int > > [read_matrix](#) (std::string file_name)
- void [debugMatrix](#) (const std::vector< std::vector< int > > vect)
- void [save_solution](#) (const std::vector< std::vector< int > > solution, std::string &outputFile)
- void [save_solution](#) (int solution, std::string &outputFile)
- void [save_solution](#) (std::string input, std::string &outputFile)
- int [Determinant](#) (std::vector< std::vector< int > > A)
- void [Add](#) (const std::vector< std::vector< int > > A, const std::vector< std::vector< int > > B, std::string &outputFile)
- void [Subtract](#) (const std::vector< std::vector< int > > A, const std::vector< std::vector< int > > B, std::string &outputFile)
- void [Multiply](#) (const std::vector< std::vector< int > > A, const std::vector< std::vector< int > > B, std::string &outputFile)
- std::vector< std::vector< int > > [Transpose](#) (const std::vector< std::vector< int > > A)
- std::vector< std::vector< int > > [Cofactor](#) (const std::vector< std::vector< int > > A)
- void [Inverse](#) (const std::vector< std::vector< int > > A, std::string &outputFile)

2.2.1 Function Documentation

2.2.1.1 Add()

```
void Add (
    const std::vector< std::vector< int > > A,
    const std::vector< std::vector< int > > B,
    std::string & outputFile )
```

The function calculates the addition of two matrix and returns a solution to the file

Parameters

<i>const</i>	std::vector<std::vector<int>>>A first matrix from file
<i>const</i>	std::vector<std::vector<int>>> B second matrix from file

Returns

std::vector<std::vector<int>>> solution solution from the add operation

2.2.1.2 Cofactor()

```
std::vector< std::vector< int > > Cofactor (
    const std::vector< std::vector< int > > A )
```

The function takes a matrix and returns the cofactor of the matrix

Parameters

<i>const</i>	std::vector<std::vector<int>>> matrix The input matrix to be transposed
--------------	-------------------------------------------------------------------------

Returns

std::vector<std::vector<int>>> solution returns the cofactor matrix

2.2.1.3 debugMatrix()

```
void debugMatrix (
    const std::vector< std::vector< int > > vect )
```

The function prints a matrix on console

Parameters

<i>const</i>	std::vector<std::vector<double>>> vect vector of vector double printed to the output
--------------	--------------------------------------------------------------------------------------

2.2.1.4 Determinant()

```
int Determinant (
    std::vector< std::vector< int > > A )
```

The function calculates the determinate of a matrix and returns a solution

Parameters

<code>std::vector<std::vector<double>></code>	A a vector of vector matrix containing the values read from file
-----------------------------------------------------------	------------------------------------------------------------------

Returns

int Determinant returns the value of the determinant

2.2.1.5 Inverse()

```
void Inverse (
    const std::vector< std::vector< int > > A,
    std::string & outputFile )
```

The function calculates the inverse of a matrix and returns a matrix of vector containing the solution

Parameters

<i>const</i>	<code>std::vector<std::vector<int>></code> matrix The input matrix
<i>std::string&</i>	outputFile pointer to the file where output solution from the operation is saved

2.2.1.6 Multiply()

```
void Multiply (
    const std::vector< std::vector< int > > A,
    const std::vector< std::vector< int > > B,
    std::string & outputFile )
```

The function multiplies two matrix and returns a matrix of vector containing the solution to the file

Parameters

<i>const</i>	<code>std::vector<std::vector<int>></code> A first Vector of matrix to be multiplied
<i>const</i>	<code>std::vector<std::vector<int>></code> B second vector of matrix to be multiplied
<i>std::string&</i>	outputFile pointer to the file where output solution from the operation is saved

2.2.1.7 read_matrix()

```
const std::vector< std::vector< int > > read_matrix (
    std::string file_name )
```

The function reads a matrix of random number from a file

Parameters

<i>const</i>	std::string& file_name pointer to the file been read from
--------------	-----------------------------------------------------------

Returns

std::vector< std::vector<double>>> read_matrix returns a vector of vector matrix containing the read matrix

Extract numbers from text

2.2.1.8 save_solution() [1/3]

```
void save_solution (
    const std::vector< std::vector< int > > solution,
    std::string & outputFile )
```

The function saves the solution form a matrix operation to a file

Parameters

<i>const</i>	std::string& file_name pointer to the file been saved to
<i>const</i>	std::vector< std::vector<double>>> solution vector of vector matrix containing the matrix to be saved to file

2.2.1.9 save_solution() [2/3]

```
void save_solution (
    int solution,
    std::string & outputFile )
```

The function saves the solution form a matrix operation to a file

Parameters

<i>double</i>	solution double with value to the printed to file
<i>std::string&</i>	outputFile pointer to the file been saved to

2.2.1.10 save_solution() [3/3]

```
void save_solution (
    std::string input,
    std::string & outputFile )
```

The function saves the solution form a matrix operation to a file

Parameters

<i>std::string</i>	file_name string with error message to the printed to file
<i>std::string</i>	&outputFile pointer to the file been saved to

2.2.1.11 Subtract()

```
void Subtract (
    const std::vector< std::vector< int > > A,
    const std::vector< std::vector< int > > B,
    std::string & outputFile )
```

The function calculates the difference of two matrix and returns a solution to the file

Parameters

<i>const</i>	std::vector<std::vector<int>> matrix1 first matrix from file
<i>const</i>	std::vector<std::vector<int>> matrix2 second matrix from file
<i>std::string&</i>	outputFile pointer to the file where output solution from the operation is saved

2.2.1.12 Transpose()

```
std::vector< std::vector< int > > Transpose (
    const std::vector< std::vector< int > > A )
```

The function takes a matrix and returns the matrix transpose to the file

Parameters

<i>const</i>	std::vector<std::vector<int>> matrix The input matrix to be transposed
--------------	------------------------------------------------------------------------

Returns

std::vector<std::vector<int>> solution returns the matrix transpose

2.3 Matrix.h

[Go to the documentation of this file.](#)

```
1 #ifndef MATRIX_H
2 #define MATRIX_H
3 #include <vector>
4
10 const std::vector<std::vector<int>> read_matrix(std::string file_name);
11
16 void debugMatrix(const std::vector<std::vector<int>> vect);
17
```

```
23 void save_solution(const std::vector< std::vector<int>» solution, std::string& outputFile);
24
30 void save_solution(int solution, std::string& outputFile);
31
37 void save_solution(std::string input, std::string& outputFile);
38
39
44 int Determinant(std::vector<std::vector<int>» A);
45
51 void Add(const std::vector<std::vector<int>» A, const std::vector<std::vector<int>» B, std::string&
    outputFile);
52
58 void Subtract(const std::vector<std::vector<int>» A, const std::vector<std::vector<int>» B, std::string&
    outputFile);
59
65 void Multiply(const std::vector<std::vector<int>» A, const std::vector<std::vector<int>» B, std::string&
    outputFile);
66
72 std::vector<std::vector<int>» Transpose(const std::vector<std::vector<int>»A);
73
79 std::vector<std::vector<int>» Cofactor(const std::vector<std::vector<int>» A);
80
81
87 void Inverse(const std::vector<std::vector<int>» A, std::string& outputFile);
88 #endif
89
```

Index

- Add
 - main.cpp, [4](#)
 - Matrix.h, [9](#)
- add
 - main.cpp, [4](#)
- Cofactor
 - main.cpp, [5](#)
 - Matrix.h, [10](#)
- cofactor
 - main.cpp, [4](#)
- convert
 - main.cpp, [5](#)
- debugMatrix
 - main.cpp, [5](#)
 - Matrix.h, [10](#)
- Determinant
 - main.cpp, [6](#)
 - Matrix.h, [10](#)
- determinant
 - main.cpp, [4](#)
- Inverse
 - main.cpp, [6](#)
 - Matrix.h, [11](#)
- inverse
 - main.cpp, [4](#)
- main
 - main.cpp, [6](#)
- main.cpp, [3](#)
 - Add, [4](#)
 - add, [4](#)
 - Cofactor, [5](#)
 - cofactor, [4](#)
 - convert, [5](#)
 - debugMatrix, [5](#)
 - Determinant, [6](#)
 - determinant, [4](#)
 - Inverse, [6](#)
 - inverse, [4](#)
 - main, [6](#)
 - Multiply, [6](#)
 - multiply, [4](#)
 - operation, [4](#)
 - random, [4](#)
 - read_matrix, [7](#)
 - save_solution, [7, 8](#)
 - Subtract, [8](#)
 - subtract, [4](#)
 - Transpose, [8](#)
 - transpose, [4](#)
- Matrix.h, [9](#)
 - Add, [9](#)
 - Cofactor, [10](#)
 - debugMatrix, [10](#)
 - Determinant, [10](#)
 - Inverse, [11](#)
 - Multiply, [11](#)
 - read_matrix, [11](#)
 - save_solution, [12](#)
 - Subtract, [13](#)
 - Transpose, [13](#)
- Multiply
 - main.cpp, [6](#)
 - Matrix.h, [11](#)
- multiply
 - main.cpp, [4](#)
- operation
 - main.cpp, [4](#)
- random
 - main.cpp, [4](#)
- read_matrix
 - main.cpp, [7](#)
 - Matrix.h, [11](#)
- save_solution
 - main.cpp, [7, 8](#)
 - Matrix.h, [12](#)
- Subtract
 - main.cpp, [8](#)
 - Matrix.h, [13](#)
- subtract
 - main.cpp, [4](#)
- Transpose
 - main.cpp, [8](#)
 - Matrix.h, [13](#)
- transpose
 - main.cpp, [4](#)