МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІСЬКА ПОЛІТЕХНІКА»

Кафедра ІСМ



**Звіт**

До розрахунково-графічної роботи

З дисципліни:

«Технології розподілених систем та паралельних обчислень»

Варіант №25

*Виконав:*

*студент групи КН-47*

Шандра Олег

*Прийняв:*

*доцент* Василюк А. С.

Львів 2018

**ЗМІСТ**

1. Завдання …………………………………………………………………………..3
2. Вступ……………………………………………………………………………....4
3. Розділ 1. Теоретична частина…………………………………………………....5
4. Розділ 2. Практичні частини....………………………………………………….10
5. Висновки………………………………………………………………………….12
6. Список використаної літератури………………………………………………..13

**ЗАВДАННЯ**

**Завдання 1. (теоретичне)**. У відповідності до варіанту (визначається згідно з номером студента в журналі викладача) розкрити теоретичні положення згідно з темою розрахункової роботи.

Варіант 25. Паралельний аналог «симплекс-методу».

**Завдання 2 (практичне).** Паралельні обчислення**.**

Використовуючи метод функціональної декомпозиції, розробити алгоритм обчислення запропонованого матрично-векторного виразу, який би враховував можливість паралельного виконання. ВСІ вхідні дані є цілими числами, більшими за нуль. ПРОВЕСТИ ВЕРИФІКАЦІЮ АЛГОРИТМУ.

На основі створеного алгоритму написати програму яка дозволяє обчислити вираз.

|  |  |  |
| --- | --- | --- |
| число | | |
| bi=7i | A1(b1+c1) | A2(B2-C2)  Cij=1/(i3+j2) |

**Завдання 3 (практичне). Розподілені системи.**

В даному розділі необхідно реалізувати систему у відповідності до технологій побудови розподілених систем, а саме необхідно:

* сформулювати поставлену задачі, вимоги до програми, що розробляється.
* проаналізувати існуючі рішення поставленої задачі та описати її особливості
* обґрунтувати вибраний метод розв'язання задачі
* описати та відобразити алгоритм, за допомогою якого буде розв'язана задача (необхідно описати кожний крок алгоритму, вказати граничні умови, дати оцінку швидкості алгоритму та об'єму необхідної пам'яті)
* реалізувати програмне рішення (при описані програми необхідно зробити співставлення частин алгоритму та фрагментів програми)
* в контрольних прикладах необхідно показати 1-2 сеанси роботи з програмою, що демонструють основні можливості роботи програми.

Варіант 25. Розподілена система обліку купівлі-продажу земельних ділянок.

**ВСТУП**

Для розв’язання багатьох задач (прогноз погоди, задачі гідро- і газодинаміки, квантової хімії, астрономії, спектроскопії, біології, ядерної фізики тощо) необхідна висока продуктивність обчислень, висока швидкість передачі інформації по каналах зв’язку та великі об’єми оперативної і постійної пам’яті. Одним з шляхів забезпечення таких вимог є організація паралельних обчислювальних процесів і відповідних технічних засобів їх реалізації. Причому, ефективність паралельної обробки залежить як від продуктивності комп’ютерів, так і від розмірів і структури пам’яті, пропускної здатності каналів зв’язку, використаних мов програмування, компіляторів, операційних систем, чисельних методів та інших математичних досліджень. Такий широкий обсяг параметрів вимагає проведення досліджень на різних рівнях: на рівні розпаралелення алгоритмів, створення спеціальних мов програмування, компіляторів, багатопроцесорних систем, неоднорідних систем, кластерів.

Для скорочення термінів розробки паралельних засобів та дослідження їх роботи використовується моделювання.

Метою виконання роботи є засвоєння основних методів та алгоритмів моделювання паралельних обчислювальних процесів, принципів побудови відповідних структур, набуття початкових практичних навиків.

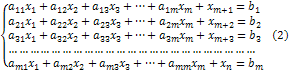
**РОЗДІЛ 1. ТЕОРЕТИЧНА ЧАСТИНА**

Ідея побудови розв'язку задачі лінійного програмування за симплекс методом полягає у переході від одного опорного плану до іншого, при якому значення цільової функції збільшується. Перехід до іншого опорного плану можливий лише в тому випадку, якщо відомо, деякий, початковий план.

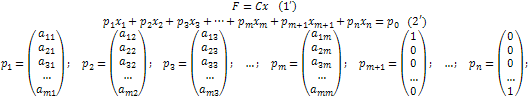
Розглянемо задачу лінійного програмування виду:

http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod4.gif

При обмеженнях:



Дану задачу запишемо у векторній формі:



Далі, виходячи з того, що http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod5.gif, можна зробити висновок, що план http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod6.gif є опорним планом задачі лінійного програмування. Цей план задається системою одиничних векторів http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod7.gif, які утворюють базис. Введемо позначення:

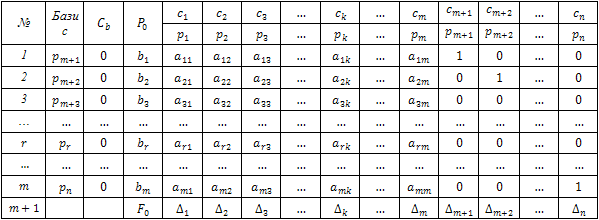
http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod30.gif

Означення 1: Опорний план http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod6.gif є оптимальним, якщо усі http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod9.gif.

Означення 2: Якщо для деякого http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod11.gif існує http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod10.gif і в http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod12.gif-му стовпці, що відповідає http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod13.gif немає додатніх елементів, то цільова функція є необмеженою на множині її планів.

Означення 3: Якщо для деякого http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod11.gif існує http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod10.gif і в http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod12.gif-му стовпці, що відповідає http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod13.gif є хоча б один додатній елемент, то існує деякий опорний план http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod14.gif, для даної задачі, в якому значення цільової функції є більше за http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod15.gif, тобто http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod16.gif.

Весь хід роботи за симплекс методом доцільно оформляти у вигляді таблиці наступного виду:



Симплекс таблиця

де значення http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod19.gif дорівнює скалярному добутку вектора http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod21.gif на вектор http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod22.gif:

http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod31.gif

Після заповнення таблиці, вихідний опорний план перевіряють на оптимальність. Для цього переглядають елементи http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod23.gif-го рядка таблиці. В результаті може мати місце один з наступних трьох випадків

Усі http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod9.gif.

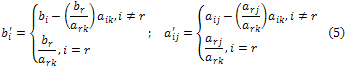
Існує http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod24.gif, для якого http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod25.gif і для кожного такого http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod24.gif симплекс таблиці, міститься принаймні одне додатнє число http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod26.gif (http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod27.gif).

Існує http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod24.gif, для якого http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod25.gif і всі відповідні цьому індексом величини http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod28.gif.

В першому випадку ми отримали оптимальний розв'язок задачі лінійного програмування. Значення http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod19.gifостанньої симплекс таблиці буде містити максимальне значенням цільової функції. В другому випадку є можливість покращити значення цільової функції за допомогою переходу до іншого опорного плану (перехід від одного опорного плану до іншого здійснюється заміною базису, тобто виключення з нього якоїсь змінної та включення замість неї нової, з числа вільних змінних). Третій випадок свідчить про необмеженість цільової функції на множині розв'язків.

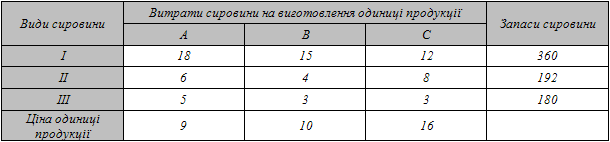
Далі розглянемо, яким чином здійснюється перехід до іншого опорного плану. Для цього серед елементів оцінкового рядка останньої симплекс таблиці вибираємо те значення http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod32.gif, яке по абсолютній величині приймає максимальне значення. Якщо їх є декілька, то вибираємо те, якому відповідає найбільше http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod33.gif. Після того, як ми вибрали http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod34.gif-й стовпець (http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod10.gif), вектор http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod35.gif потрібно ввести в базис.

Для того, щоб визначити на місце якого вектора базису вводити вектор http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod35.gif, визначаємо http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod36.gif, для усіх http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod37.gif. Нехай це буде елемент, який міститься в http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod38.gif-му рядку, тобто елемент http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod39.gif. Надалі даний елемент будемо називати розв'язуючим елементом. Стовпець http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod34.gif і рядок http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod38.gif, напрямляючими стовбцем і рядком відповідно. Наступний крок полягає у побудові нової симплекс таблиці, тобто визначення усіх її коефіцієнтів згідно нового базису, які обчислюються за наступними формулами:



Розв'язок задачі лінійного програмування за симплекс методом — приклад:

Для виготовлення товару A, B і C підприємство використовує три види сировини I, II, III. Норми витрат сировини на виробництво одного товару кожного виду, ціна одиниці товару A, B і C а також загальна кількість сировини наведені в наступній таблиці:



Норми витрат сировини на виробництво одного товару

Складемо такий план випуску даної продукції, щоб прибуток від її реалізації був максимальним.

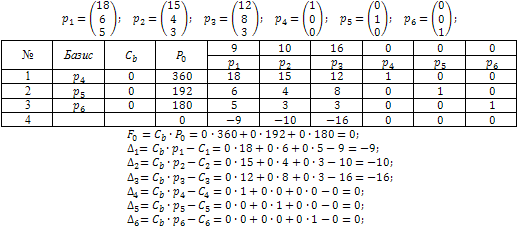
Позначемо через http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod42.gif — кількість товару А; http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod43.gif — кількість товару В; http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod44.gif — кількість товару С. Тоді математична модель даної задачі буде наступна: знайти максимум функції http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod48.gif при обмеженнях

http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod49.gif

Для побудови першого опорного плану систему нерівностей приведемо в систему рівнянь, шляхом введення додаткових змінних http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod50.gif (іншими словами запишемо систему обмежень у канонічній формі). У цільову функцію ці змінні увійдуть з нульовими коефіцієнтами:

http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod51.gif

Запишемо дану задачу у векторній формі і побудуємо першу симплекс таблицю:



Побудова першого опрного плану задачі лінійного програмування

де елементи http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod19.gif та http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod9.gif четвертого, оцінкового, рядка, обчислюються за формулами (3), (4) відповідно.

Після обчислення всіх оцінок опорний план перевіримо на оптимальність. Для цього, як уже зазначалься вище, переглядаємо елементи оцінкового рядка, бачимо, що у даному прикладі перший опорний план не є оптимальним (серед елементів http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod32.gif є такі, що мають від'ємне значення). Тому слідуючи вище розглянутому алгоритму симплекс методу, переходимо до іншого опорного плану.

Для цього серед усіх http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod32.gif вибираємо те, яке по абсолютній величині приймає максимальне значення. В нашому випадку таким буде http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod53.gif.  Тобто вектор http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod54.gif потрібно ввести в базис.

Далі, визначаємо на місце якого вектора базису вводимо http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod54.gif. Для цього визначаємо http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod55.gif. Таким чином розв'язуючим буде елемент http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod56.gif, який вказує на те, що виводити з базису необхідно вектор http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod57.gif.

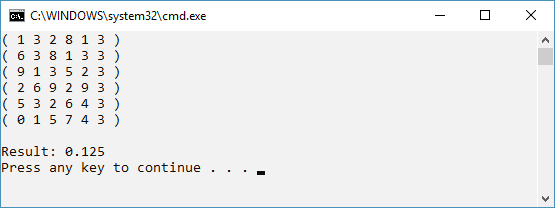
Побудова другого опрного плану задачі лінійного програмування

Після того, як ми заповнили останній (оцінковий) рядок другої симплекс таблиці, робимо висновок, що другий опорний план також не є оптимальним (серед елементів http://www.mathros.net.ua/wp-content/uploads/2012/07/simplex_metod32.gif містяться від'ємні значення). Тому, переходимо до третього опорного плану, для якого, як можна побачити нижче, умова оптимальності виконується, і який приймаємо в якості оптимального розв'язку заданої задачі лінійного програмування:

**РОЗДІЛ 2.**

**ПРАКТИЧНА ЧАСТИНА 1**

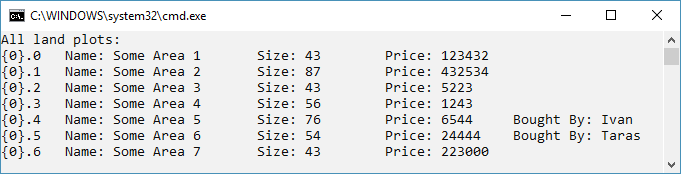
У даній практичні частині було створено сайт аплікацію яка використовує метод функціональної декомпозиції та реалізує алгоритм матрично-векторного виразу, який би враховував можливість паралельного виконання. Також було проведено верифікацію алгоритму. Аплікація створення за допомогою об’єктно орієнтованої мови програмування C#.



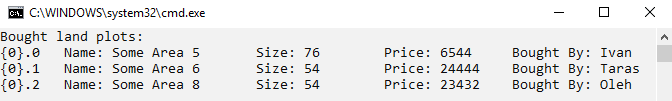
*Рис. 1. Результат виконання завдання.*

**ПРАКТИЧНА ЧАСТИНА 2**

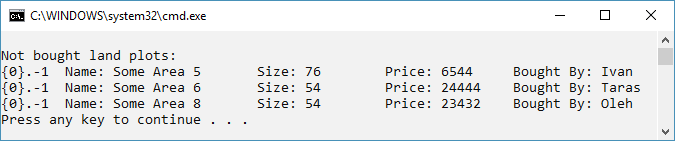
У даній практичні частині було створено сайт аплікацію що реалізує розподілену систему опліку купівлі-продажу замельних ділянок яка дозволяє переглядати куплені, не куплені та всі земельні ділянки. Аплікація створення за допомогою об’єктно орієнтованої мови програмування C#.



*Рис. 2. Саписок усіх змельних ділянок.*



*Рис. 3. Саписок куплених змельних ділянок.*



*Рис. 4. Саписок ще не куплених змельних ділянок.*

**ВИСНОВКИ**

Виконання теоретичної частино даної розрахунково-графічної роботи дало змогу ознайомитися з методом «симплекс-метод» та його паралельними реалізаціями.

У першій практичні частині було створено сайт аплікацію яка використовує метод функціональної декомпозиції та реалізує алгоритм матрично-векторного виразу, який би враховував можливість паралельного виконання. Також було проведено верифікацію алгоритму.

У другій практичні частині було створено сайт аплікацію що реалізує розподілену систему опліку купівлі-продажу замельних ділянок яка дозволяє переглядати куплені, не куплені та всі земельні ділянки.

**СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ**

1. Документація з використання ASP.NET Web API 2 (C#) [Електронний ресурс]. Режим доступу URL: https://docs.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/tutorial-your-first-web-api.
2. Офіційна документація Bootstrap. [Електронний ресурс]. Режим доступу URL: https://getbootstrap.com/.
3. Форум програмістів Електронний ресурс]. Режим доступу URL: https://stackoverflow.com/.
4. ГОСТ 34.201-90. Информационная технология. Комплекс стандартов и руководящих документов на автоматизированные системы. Автоматизированные системы**.** Стадии создания. – М.: Изд-во стандартов, 1991. – 16 с.
5. ДСТУ 3008-95. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення. – К.: Держстандарт України, 1995. – 36 с.
6. Сторінка в ресурсі Вікіпедія. [Електронний ресурс]. Режим доступу URL: https://uk.wikipedia.org/wiki/PHP

Додаток А

using System;

using System.Collections.Generic;

using System.Linq;

namespace RGP {

class LandPlot {

public string Name { get; set; }

public double Price { get; set; }

public double Size { get; set; }

public bool Bought { get; set; }

public string BoughtName { get; set; }

}

class Program {

static void Main( string [] args ) {

var landPlots = new List<LandPlot>() {

new LandPlot() { Name = "Some Area 1", Price = 123432, Size = 43 },

new LandPlot() { Name = "Some Area 2", Price = 432534, Size = 87 },

new LandPlot() { Name = "Some Area 3", Price = 5223, Size = 43 },

new LandPlot() { Name = "Some Area 4", Price = 1243, Size = 56 },

new LandPlot() { Name = "Some Area 5", Price = 6544, Size = 76, Bought = true, BoughtName = "Ivan" },

new LandPlot() { Name = "Some Area 6", Price = 24444, Size = 54, Bought = true, BoughtName = "Taras" },

new LandPlot() { Name = "Some Area 7", Price = 223000, Size = 43 },

new LandPlot() { Name = "Some Area 8", Price = 23432, Size = 54, Bought = true, BoughtName = "Oleh" },

new LandPlot() { Name = "Some Area 9", Price = 76556, Size = 23 },

};

Console.WriteLine( "All land plots:" );

landPlots.ForEach(i => {

Show( landPlots.IndexOf( i ), i );

} );

Console.WriteLine( "\nBought land plots:" );

var boughtLandPlots = landPlots.Where( i => i.Bought ).ToList();

boughtLandPlots.ForEach( i => {

Show( boughtLandPlots.IndexOf( i ), i );

} );

Console.WriteLine( "\nNot bought land plots:" );

var notBoughtLandPlots = landPlots.Where( i => !i.Bought ).ToList();

boughtLandPlots.ForEach( i => {

Show( notBoughtLandPlots.IndexOf( i ), i );

} );

}

static void Show( int index, LandPlot item) {

Console.Write( "{0}." + index );

Console.Write( "\tName: " + item.Name);

Console.Write( "\tSize: " + item.Size );

Console.Write( "\tPrice: " + item.Price );

if (item.Bought) {

Console.Write( "\tBought By: " + item.BoughtName );

}

Console.WriteLine();

} } }

Додаток Б

using System;

using System.Text;

namespace Lab4 {

class Program {

static void Main( string [] args ) {

Matrix m1 = new Matrix

( new int [,] {{1, 3, 2, 8, 1, 3},

{6, 3, 8, 1, 3, 3},

{9, 1, 3, 5, 2, 3},

{2, 6, 9, 2, 9, 3},

{5, 3, 2, 6, 4, 3},

{0, 1, 5, 7, 4, 3}} );

var d = new Decomposition();

Console.WriteLine( "Result: " + d.CalculateX( m1 ) );

}

}

class Decomposition {

private double i { get; set; }

private double j { get; set; }

private double b1 { get { return 7; } }

private double b2 { get { return 1 / Math.Pow( i, 3 ); } }

private double C { get { return 1 / Math.Pow( i + j, 3 ); } }

public double CalculateX( Matrix m ) {

i = 1;

j = 1;

return C;

}

}

/// <summary>

/// Class Matrix that contain

/// values array, it's Width and Height

/// and some static methods

/// </summary>

public class Matrix {

public Matrix( int [,] array ) {

this.array = array;

}

private int [,] array;

public int Height {

get {

return array.GetLength( 0 );

}

}

public int Width {

get {

return array.GetLength( 1 );

}

}

/// <summary>

/// Getting complementary minor

/// </summary>

/// <param name="M">Matrix search minor</param>

/// <param name="k">Minor order</param>

/// <returns>Double Get minor wiht M and k - 1 parameters</returns>

public static double GetComplementaryMinor( Matrix M, int k ) {

return GetMinor( M, k - 1 );

}

/// <summary>

/// Getting k-ordered minor

/// </summary>

/// <param name="M">Matrix search minor</param>

/// <param name="k">Minor order</param>

/// <returns>Double Get minor wiht M, k and k parameters</returns>

public static double GetMinor( Matrix matrix, int k ) {

var maxMinorRange = Math.Min( matrix.Height, matrix.Width );

var square = new int [maxMinorRange, maxMinorRange];

//Getting square from rectengle

for (int i = 0; i < maxMinorRange; i++) {

for (int j = 0; j < maxMinorRange; j++) {

square [i, j] = matrix.array [i, j];

}

}

return GetMinor( new Matrix( square ), k, k );

}

/// <summary>

/// Getting minor with removing row and collum

/// </summary>

/// <param name="M">Matrix search minor</param>

/// <param name="i">Row removed</param>

/// <param name="j">Collum removed</param>

/// <returns>Dooble Determinant</returns>

public static double GetMinor( Matrix matrix, int i, int j ) {

//In case non-square matrix

if (matrix.Height != matrix.Width) {

throw new ArgumentException( "No i- j- mionors for rectangle matrix" );

}

var size = matrix.Width;

//Matrix with removing row i

var matrixWithoutRow = new int [size - 1, size];

//Result matrix - matrix with removing row i and collum j

var resultMatrix = new int [size - 1, size - 1];

//Creating Matrix with removing row i

for (int n = 0; n < size - 1; n++) {

for (int m = 0; m < size; m++) {

if (n < i) {

matrixWithoutRow [n, m] = matrix.array [n, m];

} else {

matrixWithoutRow [n, m] = matrix.array [n + 1, m];

}

}

}

//Creating result matrix

for (int n = 0; n < size - 1; n++) {

for (int m = 0; m < size - 1; m++) {

if (m < j) {

resultMatrix [n, m] = matrixWithoutRow [n, m];

} else {

resultMatrix [n, m] = matrixWithoutRow [n, m + 1];

}

}

}

return Determinant( new Matrix( resultMatrix ) );

}

/// <summary>

/// Adding two minors

/// </summary>

/// <param name="minor1">First minor to adding</param>

/// <param name="minor2">Second minor to adding</param>

/// <returns>Result of adding</returns>

public static double AddMinors( double minor1, double minor2 ) {

return minor1 + minor2;

}

/// <summary>

/// Multiplication two minors

/// </summary>

/// <param name="minor1">First minor to multiplicate</param>

/// <param name="minor2">Second minor to multiplicate</param>

/// <returns>Result of multiplication</returns>

public static double MultiplicateMinors( double minor1, double minor2 ) {

return minor1 \* minor2;

}

/// <summary>

/// Dividing two minors

/// </summary>

/// <param name="minor1">First minor</param>

/// <param name="minor2">Second minor to divide</param>

/// <returns>Result of dividing</returns>

public static double DivideMinors( double minor1, double minor2 ) {

return minor1 / minor2;

}

/// <summary>

/// Finding determinant of Matrix

/// </summary>

/// <param name="Matrix">Target Matrix</param>

/// <returns>Double result</returns>

public static double Determinant( Matrix inputSquare ) {

//In case of one-value matrix.

if (inputSquare.Height == 1) {

return inputSquare.array [0, 0];

}

//In case of two-order matrix.

else if (inputSquare.Height == 2) {

//Calculating by mathematical formula.

return inputSquare.array [0, 0] \* inputSquare.array [1, 1]

- inputSquare.array [1, 0] \* inputSquare.array [0, 1];

}

double result = 0;

int size = inputSquare.Width;

for (int i = 0; i < size; i++) {

//Calculating by mathematical formula.

result += Math.Pow( -1, i ) \* inputSquare.array [0, i] \* GetMinor( inputSquare, 0, i );

}

return result;

}

/// <summary>

/// Converting to string for convenience presentation format

/// </summary>

/// <return>String look</return>

public override string ToString() {

StringBuilder result = new StringBuilder();

for (int i = 0; i < Height; i++) {

result.Append( "( " );

for (int j = 0; j < Width; j++) {

result.Append( array [i, j] );

result.Append( " " );

}

result.Append( ")" );

result.AppendLine();

}

return result.ToString();

}

}

}