**4 ВЕРИФІКАЦІЯ**

**4.1 ВЕРИФІКАЦІЯ**

Таблиця 4.1 – Верифікація та її результати

|  |  |  |  |
| --- | --- | --- | --- |
| Вимоги в ТЗ | Прецедент | Реалізація | Висновок |
| 1. Порівняння графів на еквівалентність |  |  | Реалізовано |
| – Функція побудування векторів повних підграфів | Порівняння графів на еквівалентність | IsEqualTo() | Реалізовано |
| 2. Графічне представлення графів |  |  | Реалізовано |
| – Функція додавання нових вершин | Додати нову вершину | AddTop() | Реалізовано |
| – Функція додавання нових ребер | Додати нове ребро | AddFirstTopOfRib()  AddSecondTopOfRib() | Реалізовано |
| – Функція видалення вершин | Видалення вершини та прилеглих ребер | DeleteTop() | Реалізовано |
| – Функція збереження поточного графу | Збереження графу у файл \*.JSON | SaveAll() | Реалізовано |
| – Функція завантаження графу з пам’яті | Завантаження графу з файлу \*.JSON | LoadAll() | Реалізовано |

**4.2 ТЕСТУВАННЯ**

Для функціонального тестування програми було побудовано графи, представлені на рисунках 4.2.1 – 4.2.4.



Рисунок 4.2.1 – Граф для тестування №1



Рисунок 4.2.2 – Граф для тестування №2



Рисунок 4.2.3 – Граф для тестування №3



Рисунок 4.2.4 – Граф для тестування №4

Для тестування еквівалентності графів, було обрано наступні пари:

1. №1 та №2
2. №1 та №3
3. №1 та №4
4. №2 та №4
5. №3 та №4

В процесі тестування ми отримали наступні результати, представлені на рисунках 4.2.5 – 4.2.14.

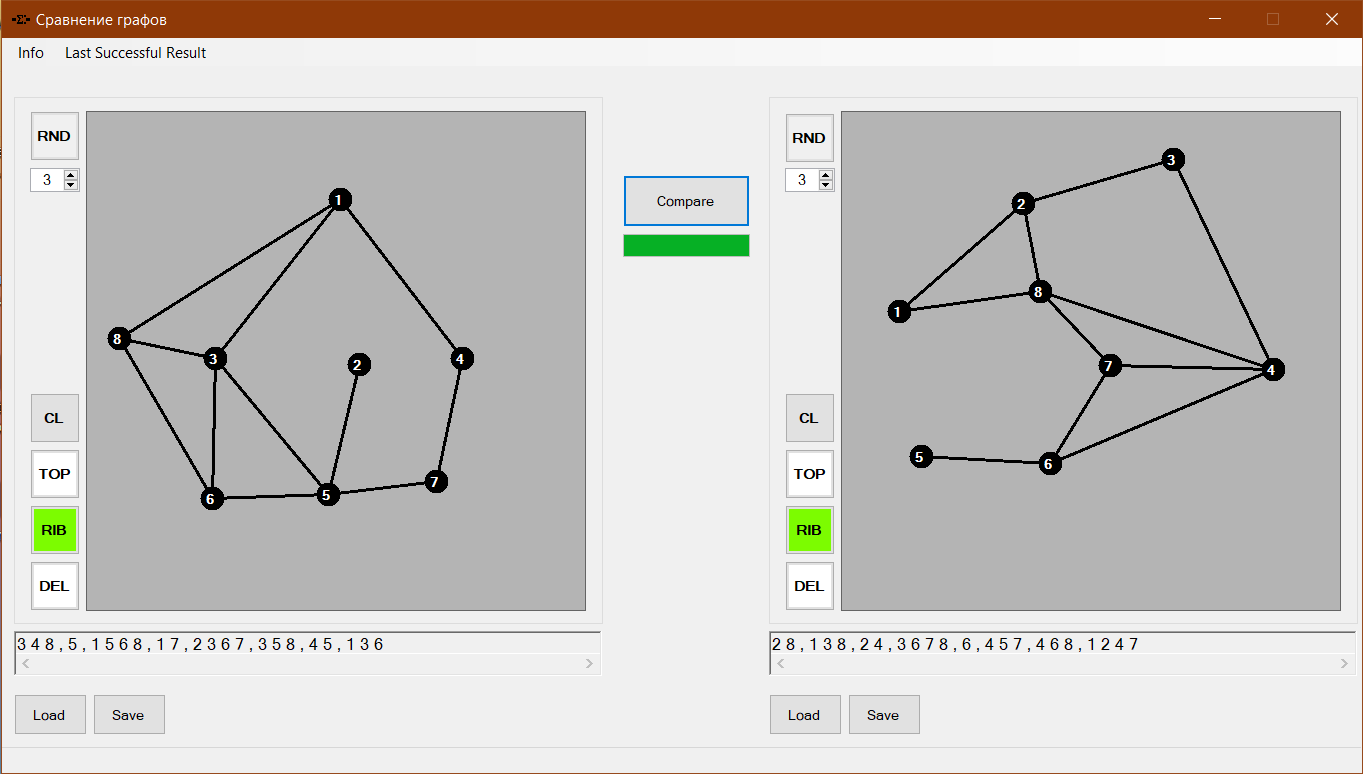


Рисунок 4.2.5 – результат зрівняння 1 та 2 графів

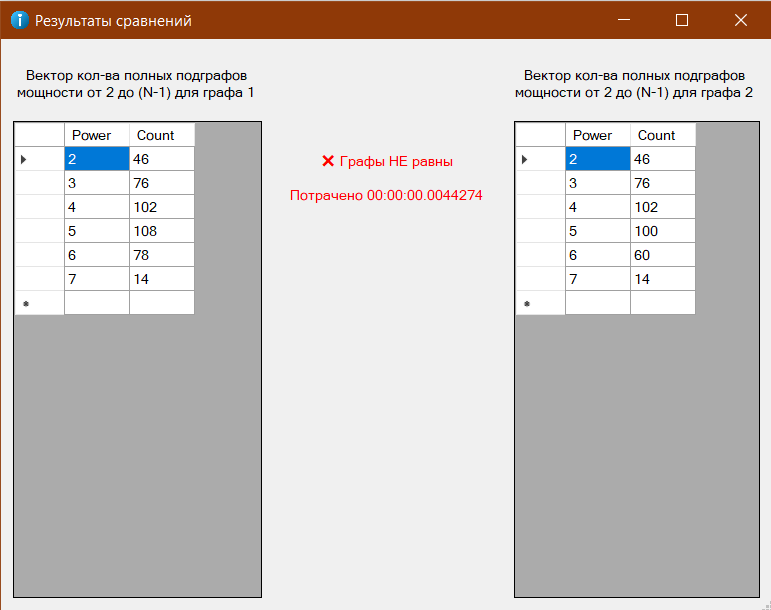


Рисунок 4.2.6 – подробиці зрівняння 1 та 2 графів

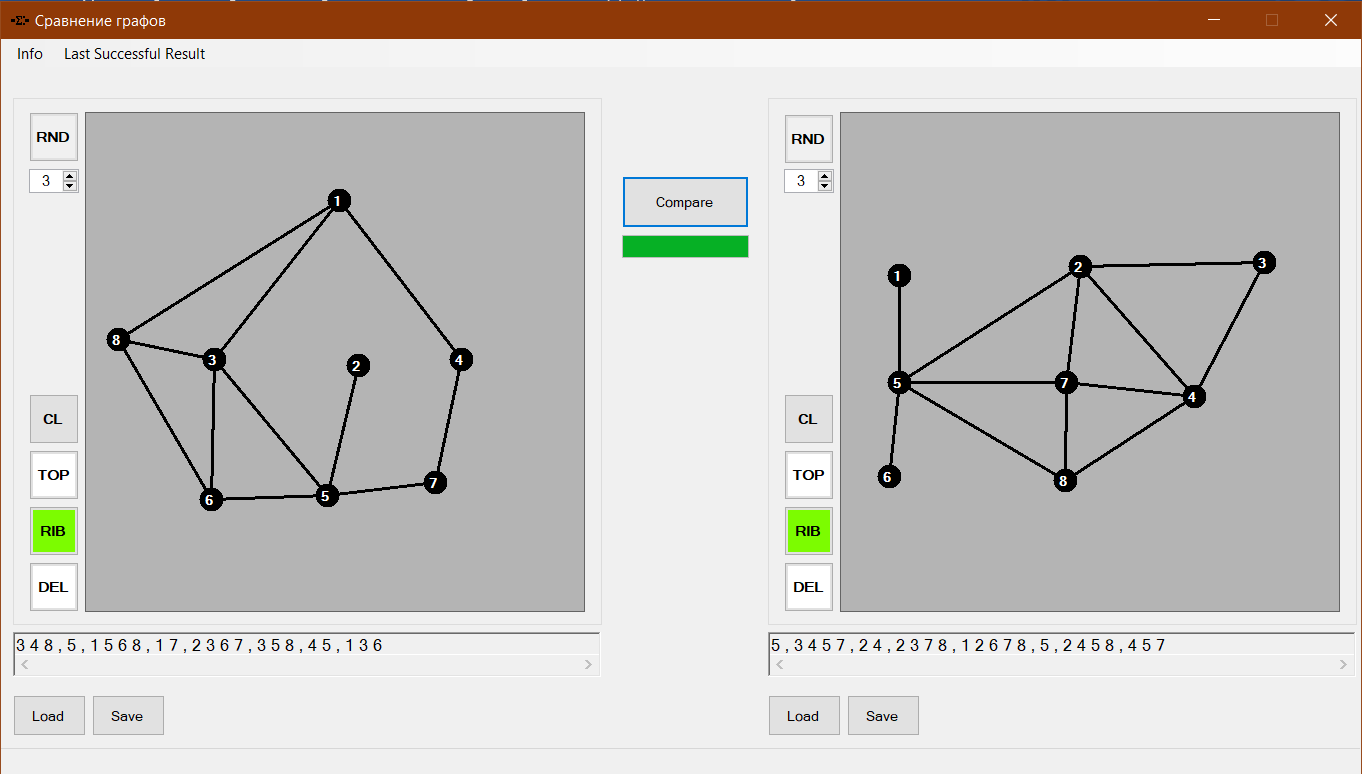


Рисунок 4.2.7 – результат зрівняння 1 та 3 графів

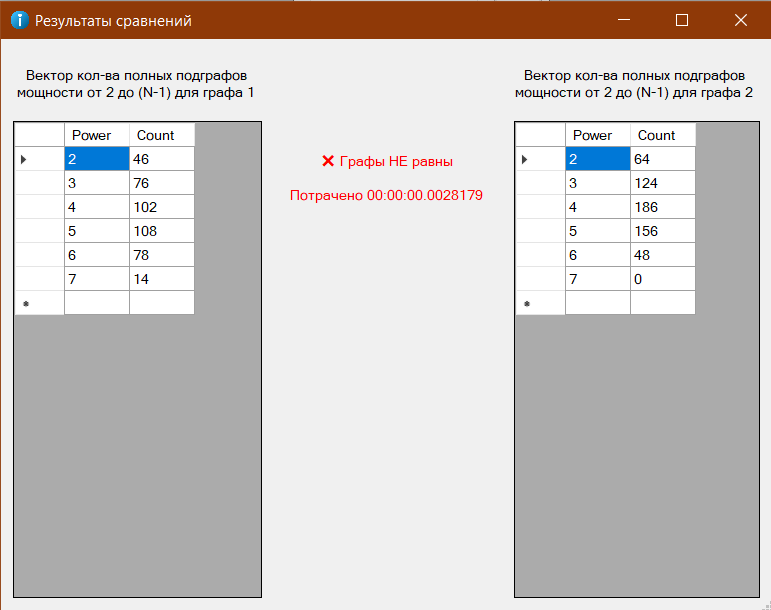


Рисунок 4.2.8 – подробиці зрівняння 1 та 3 графів

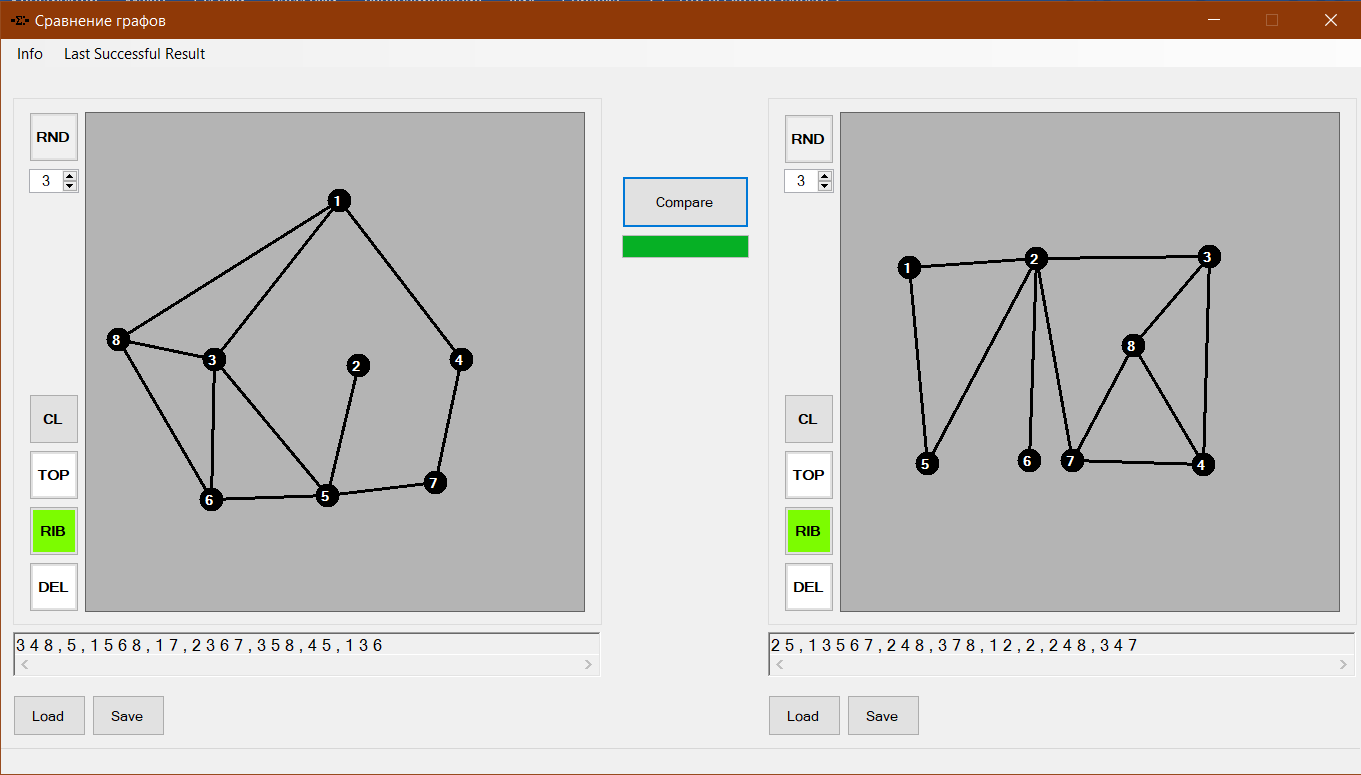


Рисунок 4.2.9 – результат зрівняння 1 та 4 графів

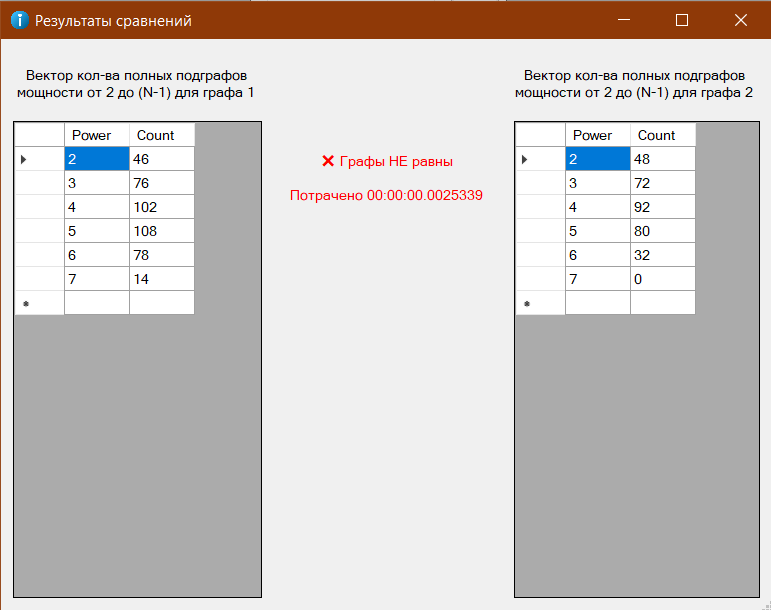


Рисунок 4.2.10 – подробиці зрівняння 1 та 4 графів

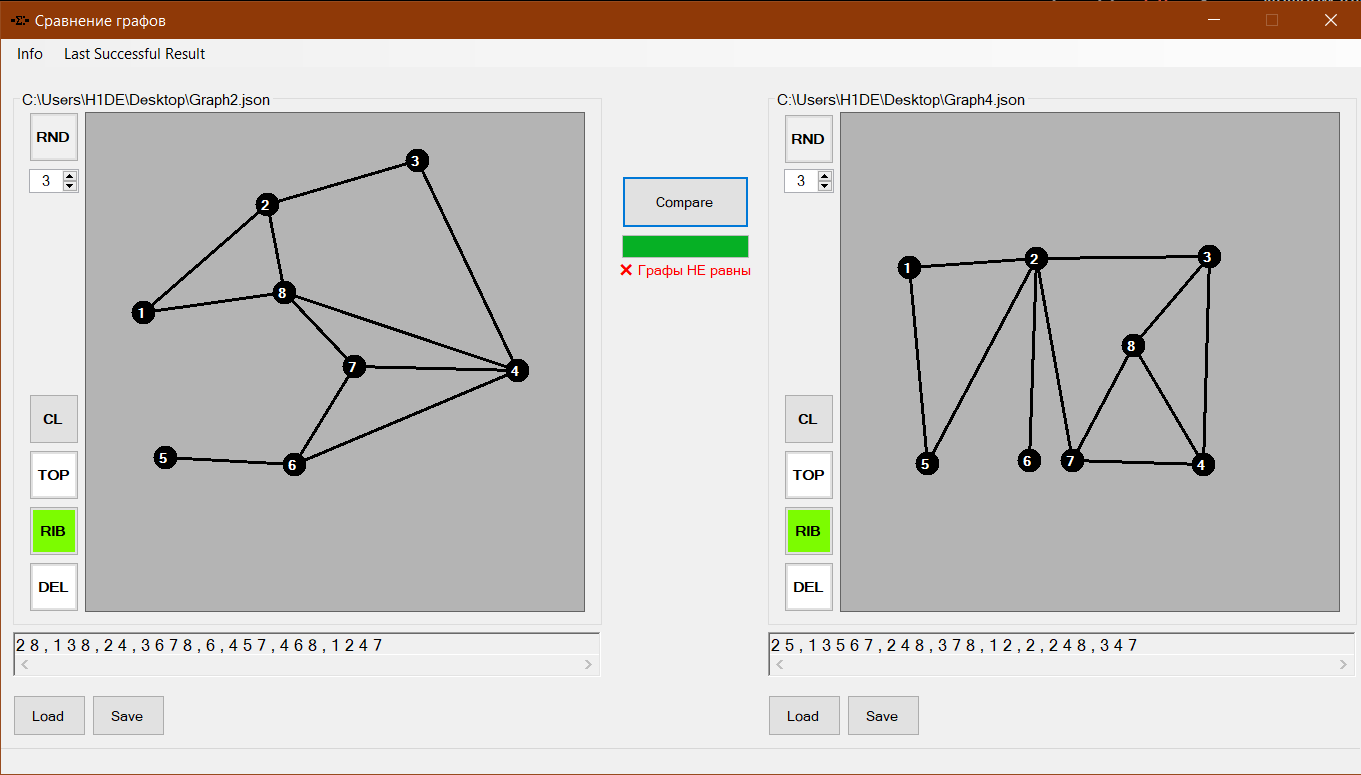


Рисунок 4.2.11 – результат зрівняння 2 та 4 графів

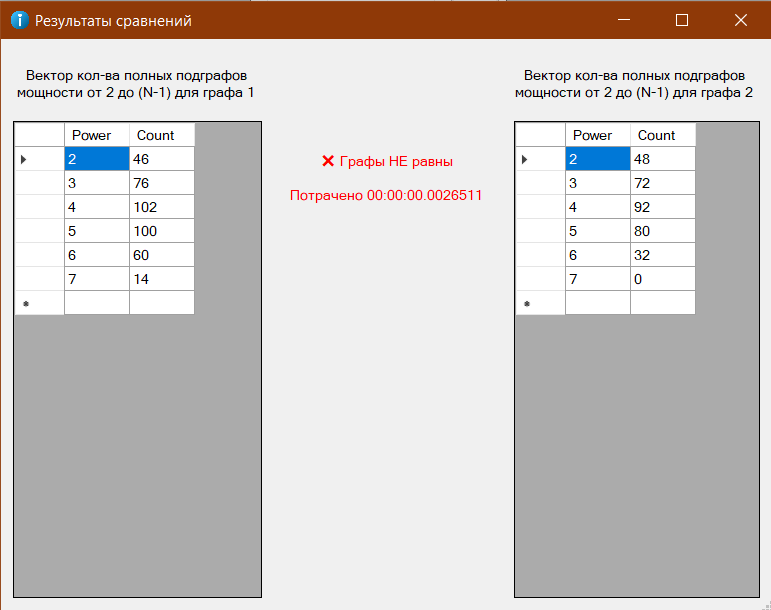


Рисунок 4.2.12 – подробиці зрівняння 2 та 4 графів

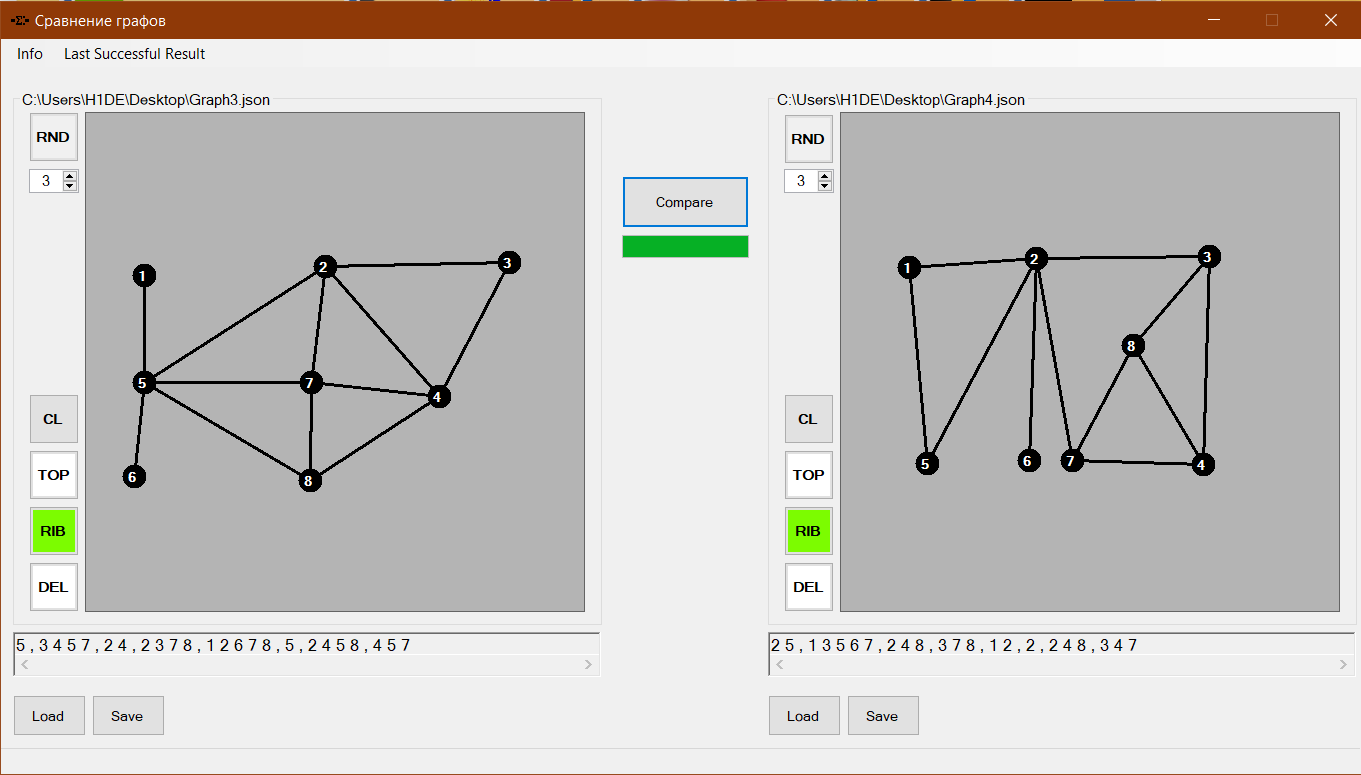


Рисунок 4.2.13 – результат зрівняння 3 та 4 графів

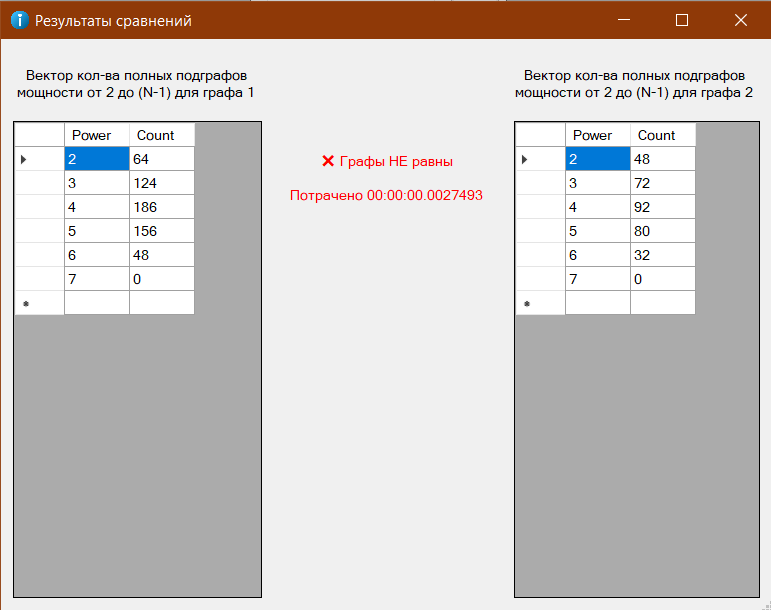


Рисунок 4.2.14 – подробиці зрівняння 3 та 4 графів

Для тестування проекту було також розроблено проект з unit - тестами найважливіших методів, а саме:

1. Генерація випадкового графу з заданою кількістю вершин;
2. Додавання ребра, у якого початок і кінець в той самій вершині;
3. Пошук вершини за від’ємними координатами;
4. Генерація FO представлення графу;
5. Порівняння графів з різною кількістю вершин;

Нижче представлений код unit – тестування та його результати

using System;

using Microsoft.VisualStudio.TestTools.UnitTesting;

using GraphLibrary;

using System.Windows.Forms;

namespace CourceProjectTests

{

[TestClass]

public class ProjectTests

{

[TestMethod]

public void Random\_Graph\_Test() // Тест генерации случайного графа

{

Random rand = new Random();

ProgressBar pr = new ProgressBar();

Graph test\_class = new Graph(pr);

test\_class.MakeRandomGraph((byte)rand.Next(3,256));

}

[TestMethod]

public void Similar\_Top\_In\_Rib\_Test() // Тест на добавление ребра, в котором начало и конец одна и та же вершина

{

Random rand = new Random();

ProgressBar pr = new ProgressBar();

Graph test\_class = new Graph(pr);

test\_class.MakeRandomGraph((byte)rand.Next(0, 256));

test\_class.AddFirstTopOfRib(1);

if (test\_class.AddSecondTopOfRib(1))

throw new Exception();

}

[TestMethod]

public void Minus\_Cordinates\_Test() // Теста на проверку поиска вершины, если даны отрицательные координаты

{

Random rand = new Random();

ProgressBar pr = new ProgressBar();

Graph test\_class = new Graph(pr);

test\_class.MakeRandomGraph((byte)rand.Next(0, 256));

test\_class.IsThereTop(-1,-1);

}

[TestMethod]

public void Make\_FO\_Test() // Тест на создание FO представления графа

{

Random rand = new Random();

ProgressBar pr = new ProgressBar();

Graph test\_class = new Graph(pr);

test\_class.MakeRandomGraph((byte)rand.Next(0, 256));

test\_class.MakeFO();

if(test\_class.GraphInFO == null)

throw new Exception();

}

[TestMethod]

public void Different\_Count\_Of\_Tops\_In\_Graph\_Test() // Проверка сравнения графов, если кол-во вершин разное

{

ProgressBar pr = new ProgressBar();

Graph test\_class1 = new Graph(pr);

Graph test\_class2 = new Graph(pr);

test\_class1.MakeRandomGraph(20);

test\_class2.MakeRandomGraph(21);

if(test\_class1.IsEqualTo(test\_class2))

throw new Exception();

}

}

}

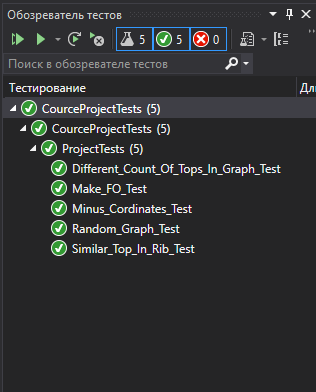


Рисунок 4.2.15 – Результати unit – тестування проекту