Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра инфокоммуникаций

**«ОСНОВЫ РАБОТЫ С ФАЙЛАМИ»**

**ОТЧЕТ**

**по лабораторной работе №10**

**дисциплины**

**«Технологии программирования»**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Выполнил:  Мизин Глеб Егорович  2 курс, группа ПИЖ-б-о-21-1,  09.03.04 «Программная инженерия», направленность (профиль) «Разработка и сопровождение программного обеспечения», очная форма обучения  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | | Проверил:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | |  | |

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_ Дата защиты\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ставрополь, 2022 г.

**Проработка учебной задачи**:

namespace Helloworld

{

    class Program

    {

        static void Main(string[] *args*)

        {

            Console.BackgroundColor = ConsoleColor.White;

            Console.ForegroundColor = ConsoleColor.Black;

            Console.Clear();

            Matrix m = new Matrix();

            m.GenerateMatrix(10,5);

            m.SaveMatrix("FileForMatrix.txt");

            if (m.LoadMatrix("FileForMatrix.txt"))

                m.PrintMatrix();

            Console.ReadKey();

        }

    }

    class Matrix

    {

        private float [,] matrix;

        int m, n;

        public Matrix()

        {

        }

        public void GenerateMatrix (int *M*, int *N*)

        {

            m = *M*; n = *N*;

            Random r = new Random(DateTime.Now.Millisecond);

            matrix = new float[*M*, *N*];

            for (int i = 0; i < *M*; i++)

                for (int j = 0; j < *N*; j++)

                    matrix[i, j] = (float)r.Next(1000) / 973f;

        }

        public void SaveMatrix(string *pFileName*)

        {

            if (matrix.Length > 0)

            {

                if (File.Exists(*pFileName*))

                    File.Delete(*pFileName*);

                FileInfo f = new FileInfo(*pFileName*);

                TextWriter tw = f.CreateText();

                tw.WriteLine(m.ToString());

                tw.WriteLine(n.ToString());

                for (int i = 0; i < m; i++)

                    for (int j = 0; j < n; j++)

                        tw.WriteLine(i.ToString() + " " + j.ToString() + " " + matrix[i, j].ToString("E10"));

                tw.Close();

            }

        }

        public Boolean LoadMatrix(string *pFileName*)

        {

            if (File.Exists(*pFileName*))

            {

                try

                {

                    TextReader tr = File.OpenText(*pFileName*);

                    m = Convert.ToInt32(tr.ReadLine());

                    n = Convert.ToInt32(tr.ReadLine());

                    matrix = new float[m, n];

                    string line;

                    string[] substring;

                    for (int i = 0; i < m; i++)

                        for (int j = 0; j < n; j++)

                        {

                            line = tr.ReadLine();

                            substring= line.Split(new char[] {' '}, 3);

                            matrix[i, j] = Convert.ToSingle(substring[2]);

                        }

                    tr.Close();

                    return true;

                }

                catch

                {

                    return false;

                }

            }

            return false;

        }

        public void PrintMatrix()

        {

            if (matrix.Length > 0)

            {

                for (int i = 0; i < m; i++)

                {

                    for (int j = 0; j < n; j++)

                        Console.Write(matrix[i, j].ToString("E3") + "  ");

                }

                Console.WriteLine();

            }

        }

    }

}

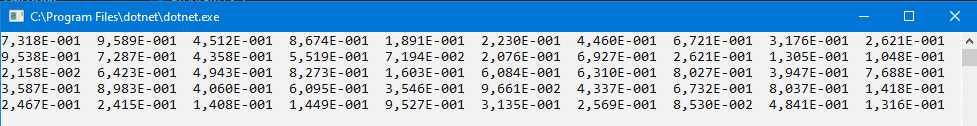


Рисунок 1 – Результат работы учебной задачи

**Индивидуальное задание**: Программа рассчитывает сумму элементов четных строк и расположенных на второстепенной диагонали в обеих матрицах, которые хранятся в разных файлах.

Код Файла Program.cs:

using StringLibrary;

namespace Helloworld

{

    class Program

    {

        static void Main(string[] *args*)

        {

            Console.BackgroundColor = ConsoleColor.White;

            Console.ForegroundColor = ConsoleColor.Black;

            Console.Clear();

            Matrix m = new Matrix();

            m.GenerateMatrix1(5,5);

            m.MatrixSave("FM1.txt");

            if (m.LoadMatrixNeededNums1("FM1.txt"))

                m.PrintMatrix1();

            m.GenerateMatrix2(3,3);

            m.MatrixSave("Filematrix2.txt");

            if(m.LoadMatrixNeededNums2("Filematrix2.txt"))

                m.PrintMatrix2();

            m.printFullSum();

            Console.ReadKey();

        }

    }

}

Для создания библиотеки класса использовалась команда «dotnet new classlib -o StringLibrary»

Код из файла созданной библиотеки Matrix.cs:

namespace StringLibrary;

public class Matrix

    {

        private int [,] matrix;

        int m, n;

        public int Sum1 = 0;

        public int Sum2 = 0;

        public int FullSum = 0;

        public Matrix()

        {

        }

        public void GenerateMatrix1 (int *M1*, int *N1*)

        {

            m = *M1*; n = *N1*;

            Random r = new Random();

            matrix = new int[*M1*, *N1*];

            for (int i = 0; i < *M1*; i++)

                for (int j = 0; j < *N1*; j++)

                    matrix[i, j] = r.Next(100, 999);

        }

        public void GenerateMatrix2 (int *M2*, int *N2*)

        {

            m = *M2*; n = *N2*;

            Random r = new Random();

            matrix = new int[*M2*, *N2*];

            for (int i = 0; i < *M2*; i++)

                for (int j = 0; j < *N2*; j++)

                    matrix[i, j] = r.Next(100, 999);

        }

        public void MatrixSave(string *pFileName*)

        {

            if (matrix.Length > 0)

            {

                if (File.Exists(*pFileName*))

                    File.Delete(*pFileName*);

                FileInfo f = new FileInfo(*pFileName*);

                TextWriter tw = f.CreateText();

                for (int i = 0; i < m; i++)

                {

                    for (int j = 0; j < n; j++)

                        tw.Write("{0} ", matrix[i, j]);

                    tw.WriteLine();

                }

                tw.Close();

            }

        }

        public Boolean LoadMatrixNeededNums1(string *pFileName*)

        {

            if (File.Exists(*pFileName*))

            {

                try

                {

                    TextReader tr = File.OpenText(*pFileName*);

                    string StrNum;

                    string[] SubStr;

                    for(int i = 0; i < n; i++ )

                    {

                    StrNum = tr.ReadLine();

                    SubStr = StrNum.Split(new char[] {' '}, n);

                    Sum1 += Convert.ToInt32(SubStr[n-i-1]);

                    if (i % 2 != 0)

                    {

                        for(int j = 0; j < n; j++)

                            if (SubStr[j] != SubStr[n-i-1])

                                Sum1 += Convert.ToInt32(SubStr[j]);

                    }

                    }

                    tr.Close();

                    return true;

                }

                catch

                {

                    return false;

                }

            }

            return false;

        }

                public Boolean LoadMatrixNeededNums2(string *pFileName*)

        {

            if (File.Exists(*pFileName*))

            {

                try

                {

                    TextReader tr = File.OpenText(*pFileName*);

                    string StrNum;

                    string[] SubStr;

                    for(int i = 0; i < n; i++ )

                    {

                    StrNum = tr.ReadLine();

                    SubStr = StrNum.Split(new char[] {' '}, n);

                    Sum2 += Convert.ToInt32(SubStr[n-i-1]);

                    if (i % 2 != 0)

                    {

                        for(int j = 0; j < n; j++)

                            if (SubStr[j] != SubStr[n-i-1])

                                Sum2 += Convert.ToInt32(SubStr[j]);

                    }

                    }

                    tr.Close();

                    return true;

                }

                catch

                {

                    return false;

                }

            }

            return false;

        }

        public void PrintMatrix1()

        {

            if (matrix.Length > 0)

            {

                for (int i = 0; i < m; i++)

                {

                    for (int j = 0; j < n; j++)

                    {

                        if (i == m-1 && j == n-1)

                        {

                            Console.Write("{0}  ", matrix[i, j]);

                            Console.WriteLine();

                        }

                        else

                        {

                            Console.Write("{0}  ", matrix[i, j]);

                        }

                    }

                    Console.WriteLine();

                }

                Console.WriteLine(Sum1);

                Console.WriteLine();

            }

        }

        public void PrintMatrix2()

        {

            if (matrix.Length > 0)

            {

                for (int i = 0; i < m; i++)

                {

                    for (int j = 0; j < n; j++)

                    {

                        if (i == m-1 && j == n-1)

                        {

                            Console.Write("{0}  ", matrix[i, j]);

                            Console.WriteLine();

                        }

                        else

                        {

                            Console.Write("{0}  ", matrix[i, j]);

                        }

                    }

                    Console.WriteLine();

                }

                Console.WriteLine(Sum2);

                Console.WriteLine();

            }

        }

        public void printFullSum()

        {

            Console.WriteLine(Sum1+Sum2);

        }

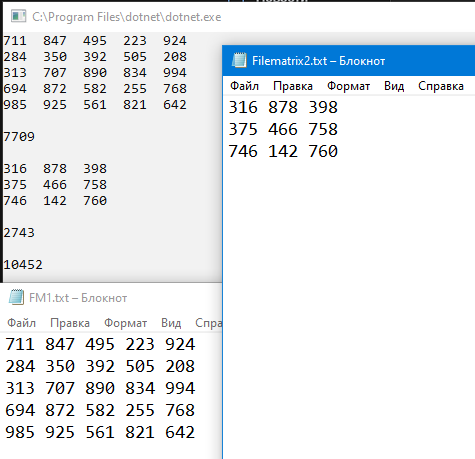


Рисунок 2 – Результат работы программы индивидуального задания

**Контрольные вопросы**:

1. Какие классы для работы с файловой системой вы знаете?

Существует 2 класса **File и FileInfo**, которые предназначены для работы с файлом, как с частью файловой системы.

2. Что такое сборка?

Сборка является базовой структурной единицей в .NET, на уровне которой проходит контроль версий, развертывание и конфигурация приложения.

3. Как определить проект по умолчанию в решении Visual Studio?

При создании нового проекта Visual Studio сохраняет его в расположении по умолчанию — **%USERPROFILE%\source\repos**. Чтобы изменить это расположение, перейдите в раздел Средства> Параметры> Проекты и решения> Расположения.

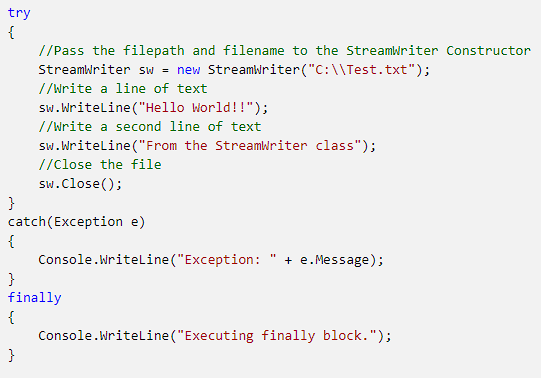
4. Какие классы отвечают за представление файлов в программе?

FileStream – представляет файл, который может быть записан, прочитан

5. Что такое поток? Какие типы классов потоков используются при работе с файлами?

Поток – это абстрактное представление последовательного устройства. Последовательное устройство – это нечто такое, что хранит данные в линейной структуре и точно таким же образом обеспечивает доступ к ним: считывает или записывает по одному байту за одну единицу времени.

6. Опишите последовательность действий при необходимости записать одну строку в файл. Приведите примеры использования различных классов.



7. Перечислите классы для работы с каталогами.

Для работы с каталогами в пространстве имен System.IO предназначены сразу два класса: **Directory и DirectoryInfo**.

8. Поясните принцип работы синтаксической конструкции try … catch.

Иногда при выполнении программы возникают ошибки, которые трудно предусмотреть или предвидеть, а иногда и вовсе невозможно. Например, при передаче файла по сети может неожиданно оборваться сетевое подключение. такие ситуации называются **исключениями**. Язык C# предоставляет разработчикам возможности для обработки таких ситуаций. Для этого в C# предназначена конструкция **try...catch...finally**.

При использовании блока **try...catch..finally** вначале выполняются все инструкции в блоке **try**. Если в этом блоке не возникло исключений, то после его выполнения начинает выполняться блок **finally**. И затем конструкция try..catch..finally завершает свою работу.

Если же в блоке try вдруг возникает исключение, то обычный порядок выполнения останавливается, и среда CLR начинает искать блок **catch**, который может обработать данное исключение. Если нужный блок catch найден, то он выполняется, и после его завершения выполняется блок finally.