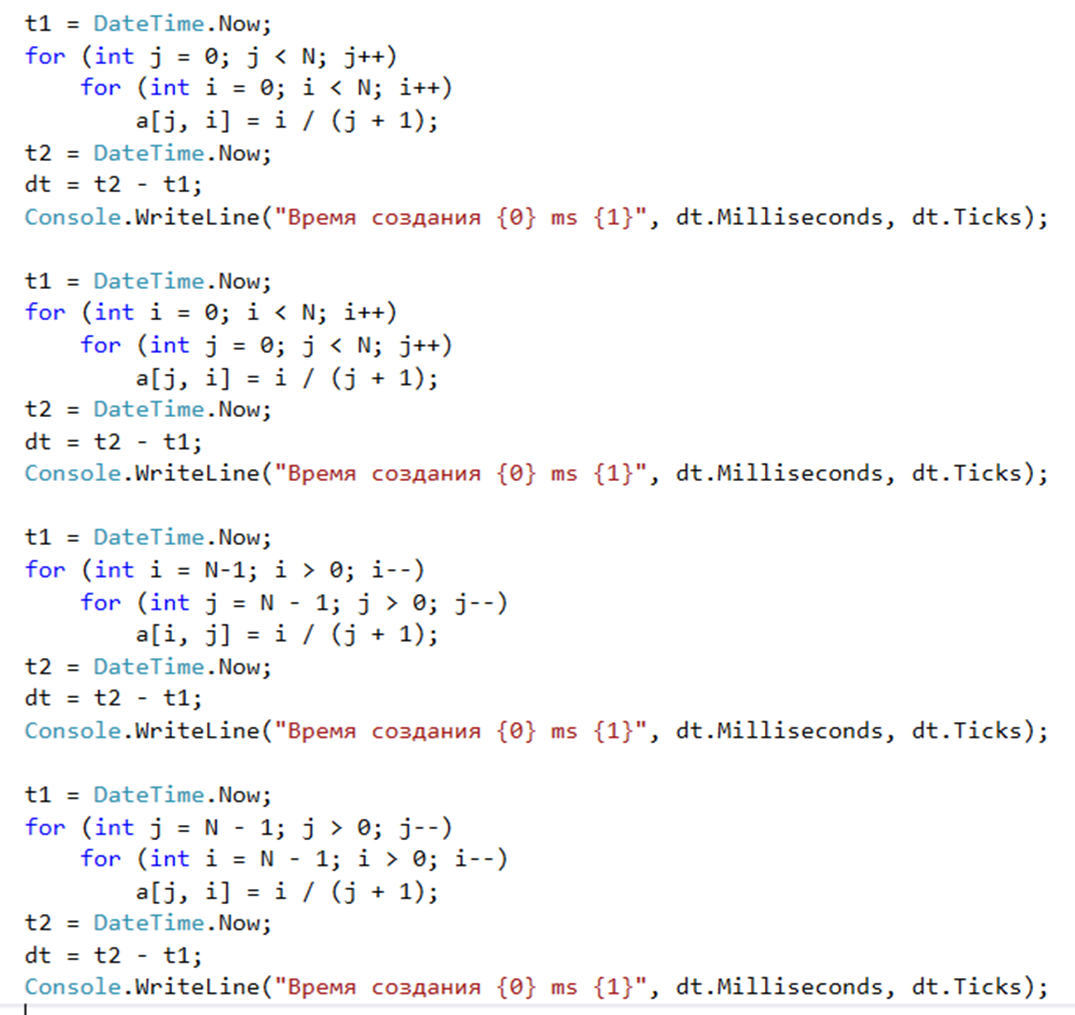
**Практическая работа №1 (10 баллов)**

**Часть 1.**

Язык программирования: предпочтительнее C# или C++ (можно выбрать другой, в котором есть реализация массивов).

Даны 4 способа перебора элементов квадратной матрицы на C#:



**вместо DateTime.Now использовать класс Stopwatch**

1.Оцените соотношение времени работы указанных 4-х способов перебора элементов квадратной матрицы:

* A(i,j),i,j=0->N,
* A(i,j),j,i=0->N,
* A(i,j),i,j=N->0,
* A(j,i),j,i=N->0.

Проведите М=1000 испытаний для N=6000. Для времени работы по каждому из способов укажите: среднее значение, min, max.

|  |  |  |  |
| --- | --- | --- | --- |
| Способ перебора | Среднее время (сек.) | Минимальное время (сек.) | Максимальное время (сек.) |
| 1 способ (A(i,j),i,j=0->N) |  |  |  |
| 2 способ (A(i,j),j,i=0->N) |  |  |  |
| 3 способ (A(i,j),i,j=N->0) |  |  |  |
| 4 способ (A(j,i),j,i=N->0) |  |  |  |

2. Проведите исследование зависимости времени работы от размерности по каждому из способов, построив график и регрессионную функцию (вид регрессии определите из трёх вариантов – линейная, квадратическая и экспоненциальная на основе МНК). Для каждой из размерностей запустить каждый способ как минимум 10 раз и учитывать при построении графика среднее значение времени для этой размерности. В качестве размерностей можно взять, например, диапазон от 10 до 10000 с шагом в 2000.

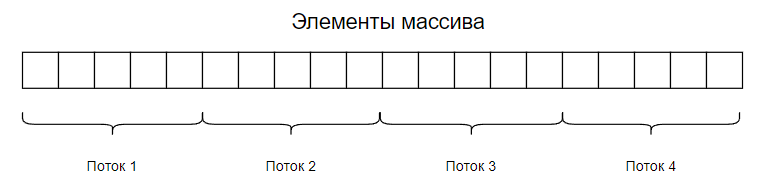
* Размерности можно скорректировать, если на больших размерностях очень долго выполняется программа.
* Графики можно отрисовать в Excel.

По двум пунктам работы сделайте выводы.

**Часть 2 (С# .NET).**

1. Даны массивы , содержащие случайные целые числа, где n – кол-во элементов массивов. Необходимо сформировать массив В, элементы которого образуются следующим образом:

* Заполнение массива В необходимо реализовать последовательно и параллельно.
* Элементы массива разделите поровну на все потоки, т. е. один поток будет обрабатывать только определенную часть массива. Например, для массива размерности 20 разделить его обработку на 4 потока:



* Запустите и определите время работы для разных размерностей и для разного количества потоков.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Размерность | Потоки | | | | | | |
| 1 | 2 | 4 | 8 | 12 | 16 | 20 |
| 1000 |  |  |  |  |  |  |  |
| 5000 |  |  |  |  |  |  |  |
| ... |  |  |  |  |  |  |  |
| 20000 |  |  |  |  |  |  |  |

* Определите нагрузку на потоки. Сколько операций выполняет каждый поток?
* Реализуйте другой способ разделения нагрузки на потоки при многопоточной обработке.
* Снова определите время работы и сравните текущие результаты с предыдущими.

Чтобы ограничить работу потоков, необходимо использовать метод Join с параметром времени. Пример:

Stopwatch sw = new Stopwatch();

sw.Start();

Thread t = new Thread(() => { Thread.Sleep(60000); });

t.Start();

if (!t.Join(new TimeSpan(0, 0, 20))) t.Abort();

sw.Stop();

Console.WriteLine(sw.ElapsedMilliseconds / 1000.0);

Вывод:

