**Федеральное агентство связи**

**Государственное бюджетное образовательное учреждение высшего**

**образование**

**Ордена Трудового Красного Знамени**

**«Московский технический университет связи и информатики»**

**Кафедра «МКиИТ»**

**дисциплина «СиАОД»**

Отчет по Лабораторной работе №1

Подготовил студент

группы БВТ1901: Балдова Татьяна

Проверил: Павликов А.

Москва 2020

# Задание 1

Реализовать заданный метод сортировки строк числовой матрицы в соответствии с индивидуальным заданием. Для всех вариантов добавить реализацию быстрой сортировки (quicksort). Оценить время работы каждого алгоритма сортировки и сравнить его со временем стандартной функции сортировки, используемой в выбранном языке программирования.

Сортировка выбором:

void ChoiceSort(int[,] ListForSort)

{

for (int k = 0; k < MatrixSize; k++)

{

int index = 0;

for (int i = 0; i < MatrixSize - 1; i++)

{

index = i;

for (int j = i + 1; j < MatrixSize; j++)

{

if (ListForSort[k, j].CompareTo(ListForSort[k, index]) == -1)

{

index = j;

}

}

if (index != i)

{

int temp = ListForSort[k, i];

ListForSort[k, i] = ListForSort[k, index];

ListForSort[k, index] = temp;

}

}

}

}

Сортировка вставками:

void InsertsSort(int[,] ListForSort)

{

for (int k = 0; k < MatrixSize; k++)

{

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

{

var temp = ListForSort[k, i];

var j = i;

while (j > 0 && temp.CompareTo(ListForSort[k, j - 1]) == -1)

{

ListForSort[k, j] = ListForSort[k, j - 1];

j--;

}

ListForSort[k, j] = temp;

}

}

}

Сортировка обменом:

void ExchangeSort(int[,] ListForSort)

{

for (int k = 0; k < MatrixSize; k++)

{

int count = MatrixSize;

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

{

for (int i = 0; i < count - 1 - j; i++)

{

var a = ListForSort[k, i];

var b = ListForSort[k, i + 1];

if (a.CompareTo(b) == 1)

{

int temp = ListForSort[k, i];

ListForSort[k, i] = ListForSort[k, i + 1];

ListForSort[k, i + 1] = temp;

}

}

count--;

}

}

}

Сортировка Шелла:

void ShellaSort(int[,] ListForSort)

{

for (int k = 0; k < MatrixSize; k++)

{

int step = MatrixSize / 2;

while (step > 0)

{

for (int i = step; i < MatrixSize; i++)

{

int j = i;

while (j >= step && ListForSort[k, j - step].CompareTo(ListForSort[k, j]) == 1)

{

int temp = ListForSort[k, j - step];

ListForSort[k, j - step] = ListForSort[k, j];

ListForSort[k, j] = temp;

j -= step;

}

}

step /= 2;

}

}

}

Быстрая сортировка:

void QuickSort(int[,] ListForSort, int left, int right, int k)

{

if (left >= right) { return; }

else

{

var pivot = Sorting(ListForSort, left, right, k);

QuickSort(ListForSort, left, pivot - 1, k);

QuickSort(ListForSort, pivot + 1, right, k);

}

}

int Sorting(int[,] ListForSort, int left, int right, int k)

{

var pointer = left;

for (int i = left; i <= right; i++)

{

if (ListForSort[k, i].CompareTo(ListForSort[k, right]) == -1)

{

int temp1 = ListForSort[k, i];

ListForSort[k, i] = ListForSort[k, pointer];

ListForSort[k, pointer] = temp1;

pointer++;

}

}

int temp = ListForSort[k, right];

ListForSort[k, right] = ListForSort[k, pointer];

ListForSort[k, pointer] = temp;

return pointer;

}

Сортировка с помощью встроенной функции:

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

{

int[] MyArray = new int[MatrixSize];

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

{

MyArray[j]=FirstMatrix[i, j];

}

Array.Sort(MyArray);

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

{

FirstMatrix[i,j]=MyArray[j];

}

}

Генерация матрицы:

int[,] CreateMatrix()

{

int[,] Matrix = new int[MatrixSize, MatrixSize];

Random rand = new Random();

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

{

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

{

Matrix[i, j] = rand.Next(-5000, 5001);

}

}

return Matrix;

}

Выбор сортировки и демонстрация результата:

//Выбор сортировки

Console.Write("1. Сортировка выбором" + "\r\n" + "2. Сортировка вставками" + "\r\n" + "3. Сортировка обменом" + "\r\n" + "4. Сортировка Шелла" + "\r\n" + "5. Сортировка быстрая" + "\r\n" + "6. Сортировка встроенная" +"\r\n"+"Выберите вид сортировки:");

//Генерация матрицы

int[,] FirstMatrix = CreateMatrix();

//Таймер для подсчета времени сортировки

Stopwatch timer = null;

//Осуществление выбранной сортировки

switch (Convert.ToInt32(Console.ReadLine()))

{

case 1:

timer = Stopwatch.StartNew();

ChoiceSort(FirstMatrix);

timer.Stop();

break;

case 2:

timer = Stopwatch.StartNew();

InsertsSort(FirstMatrix);

timer.Stop();

break;

case 3:

timer = Stopwatch.StartNew();

ExchangeSort(FirstMatrix);

timer.Stop();

break;

case 4:

timer = Stopwatch.StartNew();

ShellaSort(FirstMatrix);

timer.Stop();

break;

case 5:

timer = Stopwatch.StartNew();

for (int k = 0; k < MatrixSize; k++)

{

QuickSort(FirstMatrix, 0, MatrixSize - 1, k);

}

timer.Stop();

break;

case 6:

timer = Stopwatch.StartNew();

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

{

timer.Start();

int[] MyArray = new int[MatrixSize];

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

{

MyArray[j] = FirstMatrix[i, j];

}

Array.Sort(MyArray);

timer.Stop();

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

{

FirstMatrix[i, j] = MyArray[j];

}

}

break;

}

//Вывод времени сортировки

Console.WriteLine("Время сортировки = "+Convert.ToString(timer.ElapsedMilliseconds)+" мс");

//Вывод отсортированной матрицы

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

{

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

{

Console.Write(Convert.ToString(FirstMatrix[i, j]+" "));

}

Console.Write("\r\n");

}

Console.ReadLine();

В результате тестирования было подсчитано время, затраченное на сортировки:

|  |  |
| --- | --- |
| Вид сортировки | Время (матрица 1000 \* 1000) |
| Сортировка выбором | 7276 мс (7,276 cек) |
| Сортировка вставками | 4873 мс (4,873 cек) |
| Сортировка обменом | 7253 мс (7,253 cек) |
| Сортировка Шелла | 512 мс (0,512 сек) |
| Сортировка быстрая | 458 мс (0,458 сек) |
| Сортировка встроенная | 62 мс (0,062 сек) |

Пример отображения результата:

