Лаборатнорная работа 1

*Алгоритмы внутренней сортировки*.

ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ:

Написать программу, состоящую из следующих пунктов (в зависимости от варианта):

1. Линейная сортировка
2. Сортировка выбором
3. Поразрядная сортировка
4. Быстрая сортировка
5. Характеристики сортировок;
6. Выход.

В пп. 1- 5 продемонстрировать работу соответствующей сортировки с различными типами данных.

В п.6 для заданного типа данных и количества подсчитать время сортировки, число сравнений и перестановок ( сдвигов ) для каждого из алгоритмов.

Входные данные: чтение данных осуществляется из файла;

Выбор типа данных:

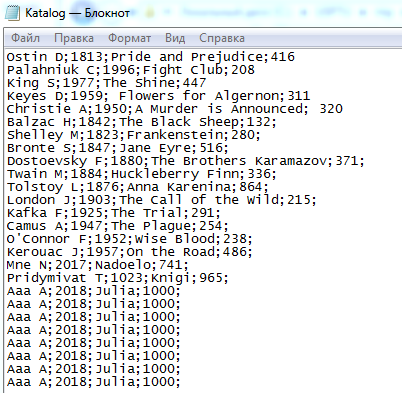
1. массив записей; ( количество 5, 10, 25)
2. массив чисел ( количество 10, 100, 1000, 10000).

Выходные данные: В пп. 1- 5 результат выводится на экран;

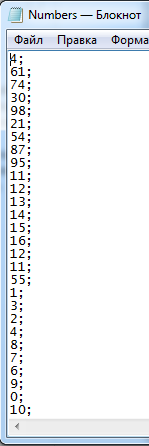
В п. 6 результат записывается в файл и считывается из него на экран в виде таблицы.

Входные файлы:

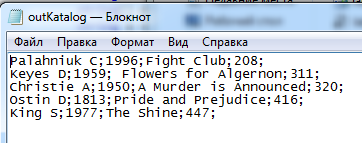
Katalog.csv

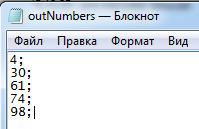


Numbers.csv

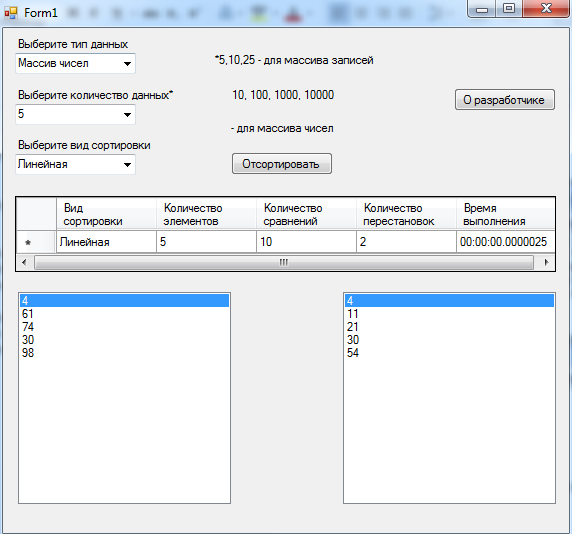


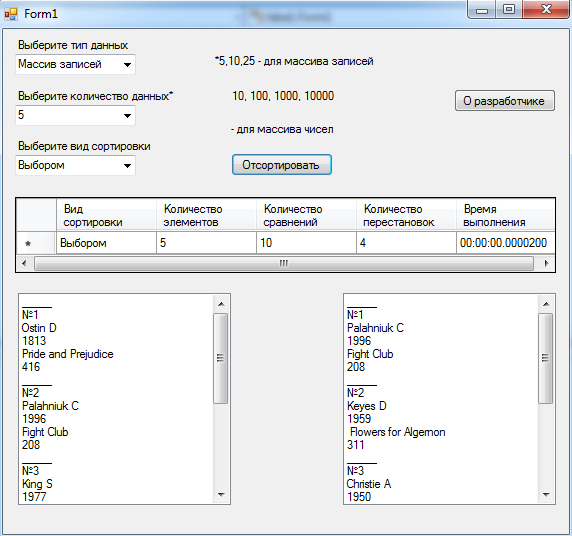
Пример выходных файлов на 5 элементов:



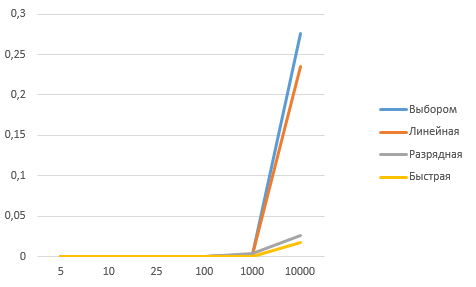


Пример работы графического интерфейса:





Графики зависимости времени сортировок от кол-ва элементов на примере массива чисел:



Вывод:

По графику мы видим, что для маленького количества элементов нет разницы, какую сортировку использовать, во времени сортировок разница несущественна. Но при большом количестве сортируемых элементов лидирует быстрый метод сортировки.

Приложение 1

Код программы

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using System.Collections;

using System.Diagnostics;

namespace laba1

{

public partial class Form1 : Form

{

struct Katalog

{

public string authorName;

public int year;

public string bookName;

public int str;

public Katalog(string authorName, int year, string bookName, int str)

{

this.authorName = authorName;

this.year = year;

this.bookName = bookName;

this.str = str;

}

}

int sr = 0, pr = 0;

void razrNot(Katalog[] arr, long first, long last)

{

double p = arr[(last - first) / 2 + first].str;

int temp, temp3;

string temp1, temp2;

long i = first, j = last;

while (i <= j)

{

while (arr[i].str < p && i <= last) ++i;

while (arr[j].str > p && j >= first) --j;

if (i <= j)

{

temp = arr[i].str;

temp1 = arr[i].authorName;

temp2 = arr[i].bookName;

temp3 = arr[i].year;

arr[i].str = arr[j].str;

arr[i].authorName = arr[j].authorName;

arr[i].bookName = arr[j].bookName;

arr[i].year = arr[j].year;

arr[j].str = temp;

arr[j].authorName = temp1;

arr[j].bookName = temp2;

arr[j].year = temp3;

++i; --j; pr++;

}

}

if (j > first) { razrNot(arr, first, j); sr++; }

if (i < last) { razrNot(arr, i, last); sr++; }

dataGridView1.Rows[0].Cells[2].Value = sr.ToString();

dataGridView1.Rows[0].Cells[3].Value = pr.ToString();

sr = 0; pr = 0;

}

void razrN(int[] arr, long first, long last)

{

double p = arr[(last - first) / 2 + first];

int temp;

long i = first, j = last;

while (i <= j)

{

while (arr[i] < p && i <= last) ++i;

while (arr[j] > p && j >= first) --j;

if (i <= j)

{

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

++i; --j; pr++;

}

}

if (j > first) { razrN(arr, first, j); sr++; }

if (i < last) { razrN(arr, i, last); sr++; }

dataGridView1.Rows[0].Cells[2].Value = sr.ToString();

dataGridView1.Rows[0].Cells[3].Value = pr.ToString();

sr = 0; pr = 0;

}

int partN(int[] array, int start, int end)

{

int marker = start;

for (int i = start; i <= end; i++)

{

sr++;

if (array[i] <= array[end])

{

int temp = array[marker];

array[marker] = array[i];

pr++;

array[i] = temp;

marker += 1;

}

}

return marker - 1;

}

void qwikN(int[] array, int start, int end)

{

if (start >= end)

{

return;

}

int pivot = partN(array, start, end);

qwikN(array, start, pivot - 1);

qwikN(array, pivot + 1, end);

}

int partN1(Katalog[] array, int start, int end)

{

int marker = start;

for (int i = start; i <= end; i++)

{

sr++;

if (array[i].str <= array[end].str)

{

int temp = array[marker].str;

string temp1 = array[marker].authorName;

string temp2 = array[marker].bookName;

int temp3 = array[marker].year;

array[marker] = array[i];

array[i].str = temp;

array[i].authorName = temp1;

array[i].bookName = temp2;

array[i].year = temp3;

marker += 1;

pr++;

}

}

return marker - 1;

}

void qwikN1(Katalog[] array, int start, int end)

{

if (start >= end)

{

return;

}

int pivot = partN1(array, start, end);

qwikN1(array, start, pivot - 1);

qwikN1(array, pivot + 1, end);

}

void lineSort(Katalog[] arr)

{

int pos = 0;

int tempIndex, temp1, temp4;

string temp2, temp3;

while (pos < arr.Length)

{

tempIndex = searchMin(arr, pos);

temp1 = arr[pos].str;

arr[pos].str = arr[tempIndex].str;

arr[tempIndex].str = temp1;

temp2 = arr[pos].authorName;

arr[pos].authorName = arr[tempIndex].authorName;

arr[tempIndex].authorName = temp2;

temp3 = arr[pos].bookName;

arr[pos].bookName = arr[tempIndex].bookName;

arr[tempIndex].bookName = temp3;

temp4 = arr[pos].year;

arr[pos].year = arr[tempIndex].year;

arr[tempIndex].year = temp4;

pos++; pr++;

}

}

int searchMin(Katalog[] arr, int pos)

{

int index = pos;

for (int i = pos + 1; i < arr.Length; ++i)

{

sr++;

if (arr[index].str > arr[i].str)

{

index = i;

}

}

return index;

}

public Form1()

{

InitializeComponent();

typeBox.Items.Add("Массив записей");

typeBox.Items.Add("Массив чисел");

qntBox.Items.Add("5");

qntBox.Items.Add("10");

qntBox.Items.Add("25");

qntBox.Items.Add("100");

qntBox.Items.Add("1000");

qntBox.Items.Add("10000");

sortBox.Items.Add("Линейная");

sortBox.Items.Add("Выбором");

sortBox.Items.Add("Разрядная");

sortBox.Items.Add("Qwik");

}

private void infoBtn\_Click(object sender, EventArgs e)

{

InfoForm f = new laba1.InfoForm();

f.Hide();

f.Show();

}

private void typeBox\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void runBtn\_Click(object sender, EventArgs e)

{

Stopwatch watch = new Stopwatch();

int iSort, iMas, iNum;

int i, k, n = 0;

iSort = sortBox.SelectedIndex;

iMas = typeBox.SelectedIndex;

iNum = qntBox.SelectedIndex;

if (iNum == 0) n = 5;

if (iNum == 1) n = 10;

if (iNum == 2) n = 25;

if (iNum == 3) n = 100;

if (iNum == 4) n = 1000;

if (iNum == 5) n = 10000;

int[] numMas = new int[n];

dataGridView1.Rows[0].Cells[0].Value = sortBox.SelectedItem;

dataGridView1.Rows[0].Cells[1].Value = qntBox.SelectedItem;

if (iMas == 0)

{//массив записей

if (!File.Exists("...\\Katalog.csv"))

{

System.Console.WriteLine("Ошибка при загрузке записей каталога: файл отсутствует!");

}

Katalog[] st = new Katalog[n];

using (StreamReader reader = File.OpenText("...\\Katalog.csv"))

{

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

{

string text = reader.ReadLine();

string[] elems = text.Split(';');

string a = (elems[0]);

int y = int.Parse(elems[1]);

string b = (elems[2]);

int s = int.Parse(elems[3]);

st[i].authorName = a;

st[i].year = y;

st[i].bookName = b;

st[i].str = s;

listBox1.Items.Add("\_\_\_\_\_");

int g = i + 1;

listBox1.Items.Add("№" + g);

listBox1.Items.Add(st[i].authorName);

listBox1.Items.Add(st[i].year);

listBox1.Items.Add(st[i].bookName);

listBox1.Items.Add(st[i].str);

}

}

if (iSort == 0)

{//Линейная

watch.Start();

lineSort(st);

watch.Stop();

TimeSpan tSpan;

tSpan = watch.Elapsed;

dataGridView1.Rows[0].Cells[4].Value = tSpan.ToString();

dataGridView1.Rows[0].Cells[2].Value = sr.ToString();

dataGridView1.Rows[0].Cells[3].Value = pr.ToString();

sr = 0; pr = 0;

}

if (iSort == 1)

{//Выбором

watch.Start();

for (i = 0; i < n - 1; i++)

{

int min = i;

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

{

sr++;

if (st[j].str < st[min].str)

{

min = j;

}

}

Katalog[] st1 = new Katalog[n];

st1[0].str = st[i].str;

st1[0].authorName = st[i].authorName;

st1[0].bookName = st[i].bookName;

st1[0].year = st[i].year;

st[i].str = st[min].str;

pr++;

st[i].authorName = st[min].authorName;

st[i].bookName = st[min].bookName;

st[i].year = st[min].year;

st[min].str = st1[0].str;

st[min].authorName = st1[0].authorName;

st[min].bookName = st1[0].bookName;

st[min].year = st1[0].year;

}

watch.Stop();

TimeSpan tSpan;

tSpan = watch.Elapsed;

dataGridView1.Rows[0].Cells[4].Value = tSpan.ToString();

dataGridView1.Rows[0].Cells[2].Value = sr.ToString();

dataGridView1.Rows[0].Cells[3].Value = pr.ToString();

sr = 0; pr = 0;

}

if (iSort == 2)

{//разрядная

watch.Start();

razrNot(st, 0, n - 1);

watch.Stop();

TimeSpan tSpan;

tSpan = watch.Elapsed;

dataGridView1.Rows[0].Cells[4].Value = tSpan.ToString();

}

if (iSort == 3)

{//qwik

watch.Start();

qwikN1(st, 0, n - 1);

watch.Stop();

TimeSpan tSpan;

tSpan = watch.Elapsed;

dataGridView1.Rows[0].Cells[4].Value = tSpan.ToString();

dataGridView1.Rows[0].Cells[2].Value = sr.ToString();

dataGridView1.Rows[0].Cells[3].Value = pr.ToString();

sr = 0; pr = 0;

}

using (StreamWriter sw = new StreamWriter("outKatalog.csv", true, Encoding.UTF8))

{

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

{

sw.Write(st[i].authorName);

sw.Write(";");

sw.Write(st[i].year);

sw.Write(";");

sw.Write(st[i].bookName);

sw.Write(";");

sw.Write(st[i].str);

sw.Write(";");

sw.WriteLine();

}

}

using (StreamReader reader2 = File.OpenText("outKatalog.csv"))

{

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

{

string text = reader2.ReadLine();

string[] elems = text.Split(';');

string a = (elems[0]);

int y = int.Parse(elems[1]);

string b = (elems[2]);

int s = int.Parse(elems[3]);

st[i].authorName = a;

st[i].year = y;

st[i].bookName = b;

st[i].str = s;

listBox2.Items.Add("\_\_\_\_\_");

int g = i + 1;

listBox2.Items.Add("№" + g);

listBox2.Items.Add(a);

listBox2.Items.Add(y);

listBox2.Items.Add(b);

listBox2.Items.Add(s);

}

}

}

if (iMas == 1)

{//массив чисел

if (!File.Exists("...\\Numbers.csv"))

{

System.Console.WriteLine("Ошибка при загрузке: файл c числами отсутствует!");

}

using (StreamReader reader1 = File.OpenText("...\\Numbers.csv"))

{

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

{

string text = reader1.ReadLine();

string[] elems = text.Split(';');

numMas[i] = int.Parse(elems[0]);

}

}

listBox1.DataSource = numMas;

if (iSort == 0)

{//Линейная

int min, tempL;

watch.Start();

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

{

min = numMas[i];

tempL = i;

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

{

sr++;

if (min > numMas[j])

{

min = numMas[j];

tempL = j;

pr++;

}

}

k = numMas[i];

numMas[i] = min;

numMas[tempL] = k;

}

watch.Stop();

TimeSpan tSpan;

tSpan = watch.Elapsed;

dataGridView1.Rows[0].Cells[4].Value = tSpan.ToString();

dataGridView1.Rows[0].Cells[2].Value = sr.ToString();

dataGridView1.Rows[0].Cells[3].Value = pr.ToString();

sr = 0; pr = 0;

}

if (iSort == 1)

{//Выбором

watch.Start();

for (i = 0; i < n - 1; i++)

{

int min = i;

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

{

sr++;

if (numMas[j] < numMas[min])

{

min = j;

}

}

pr++;

int tempV = numMas[i];

numMas[i] = numMas[min];

numMas[min] = tempV;

}

watch.Stop();

TimeSpan tSpan;

tSpan = watch.Elapsed;

dataGridView1.Rows[0].Cells[4].Value = tSpan.ToString();

dataGridView1.Rows[0].Cells[2].Value = sr.ToString();

dataGridView1.Rows[0].Cells[3].Value = pr.ToString();

sr = 0; pr = 0;

}

if (iSort == 2)

{ //разрядная

watch.Start();

razrN(numMas, 0, n - 1);

watch.Stop();

TimeSpan tSpan;

tSpan = watch.Elapsed;

dataGridView1.Rows[0].Cells[4].Value = tSpan.ToString();

dataGridView1.Rows[0].Cells[2].Value = sr.ToString();

dataGridView1.Rows[0].Cells[3].Value = pr.ToString();

sr = 0; pr = 0;

}

if (iSort == 3)

{ //qwik

watch.Start();

qwikN(numMas, 0, n - 1);

watch.Stop();

TimeSpan tSpan;

tSpan = watch.Elapsed;

dataGridView1.Rows[0].Cells[4].Value = tSpan.ToString();

}

using (StreamWriter sw = new StreamWriter("outNumbers.csv", true, Encoding.UTF8))

{

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

{

sw.Write(numMas[i]);

sw.Write(";");

sw.WriteLine();

}

}

using (StreamReader reader2 = File.OpenText("outNumbers.csv"))

{

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

{

string text = reader2.ReadLine();

string[] elems = text.Split(';');

listBox2.Items.Add(int.Parse(elems[0]));

}

}

}

}

}

}