Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э. Баумана

(национальный исследовательский университет)»

МГТУ им. Н.Э. Баумана

Факультет “Радиоэлектронные системы и комплексы”

Кафедра РЛ6 “Технологии приборостроения”

ДОМАШНЕЕ ЗАДАНИЕ № 1

по курсу “Основы программирования на C++”

Анализ алгоритмов сортировок

Студент: Филимонов С.

Группа: РЛ6-21

Преподаватель: Семеренко Д. А.

Москва

2020/2021 учебный год

Глава 1

Код программы

main.cpp

```
#include <QCoreApplication>
#include "sortirovka.h"

int main(){
  sortirovka ob1;
  return 0;
}
```

sortirovka.h

```
#ifndef SORTIROVKA_H
#define SORTIROVKA_H
class sortirovka{
private:
   int N =100000;
public:
   sortirovka();
   int pyzirek(int N);
   int vstavka(int N);
   int vstqsortRecursiveavka(int N);
   void qsortRecursive(int *mas, int size);
   int vibor(int N);
   int sorto(int N);
   int Gnome_sort(int N);
   ~sortirovka();
};
#endif
```

sortirovka.cpp

```
#include "sortirovka.h"
#include <QCoreApplication>
#include <QTextStream>
#include <QIODevice>
#include <QFile>
#include <QTime>
#include <QDebug>
#include <iostream>
#include <algorithm>
#include <utility>
#include <ctime>
#include <list>
using namespace std;
sortirovka::sortirovka(){
    qDebug() << "The start\n";
    QFile file1("D:\\Rezult Algotim\\pyzirek.txt");
    QTextStream out1(&file1);
    file1.open(QIODevice::WriteOnly);
    qDebug() << " pyzirek \n";
    for(int X = 0;X<=N;X=X+1000){
        int a = pyzirek(X);
        out1 << a << " \n";
        cout << "=";
        if(a>1000)
            break;
    }
    file1.close();
    qDebug() << "\n";
    QFile file2("D:\\Rezult Algotim\\vstavka.txt");
    QTextStream out2(&file2);
    file2.open(QIODevice::WriteOnly);
    qDebug() << " vstavka\n";
    for(int X = 0;X<=N;X=X+1000){
        int a = vstavka(X);
        out2 << a << " \n";
        cout << "=";
        if(a>1000)
            break;
    }
    file2.close();
    qDebug() << "\n";
    QFile file3("D:\\Rezult Algotim\\qsortRecursive.txt");
    QTextStream out3(&file3);
    file3.open(QIODevice::WriteOnly);
    qDebug() << " qsortRecursive\n";
    for(int X = 0;X<=N;X=X+1000){
        int a = vstqsortRecursiveavka(N);
        out3 << a << " \n";
        cout << "=";
        if(a>1000)
            break;
    }
    file3.close();
```

```
    qDebug() << "\n";
    QFile file4("D:\\Rezult Algotim\\vibor.txt");
    QTextStream out4(&file4);
    file4.open(QIODevice::WriteOnly);
    qDebug()<< " vibor\n";
    for(int X = 0;X<=N;X=X+1000){
        int a = vibor(X);
        out4 << a << " \n";
        cout << "=";
        if(a>1000)
            break;
    }
```

```
    qDebug() << "\n";
    QFile file5("D:\\Rezult Algotim\\sorto.txt");
    QTextStream out5(&file5);
    file5.open(QIODevice::WriteOnly);
    qDebug() << " sorto\n";
    for(int X = 0;X<=N;X=X+1000){
        int a = sorto(X);
        out5 << a << " \n";
        cout << "=";
        if(a>1000)
            break;
    }
    file5.close();
    qDebug() << "\n";
    QFile file6("D:\\Rezult Algotim\\gnome.txt");
    QTextStream out6(&file6);
    file6.open(QIODevice::WriteOnly);
    qDebug() << " gnome_sort\n";
    for(int X = 0;X<=N;X=X+1000){
        int a = Gnome_sort(X);
        out6 << a << " \n";
        cout << "=";
        if(a>1000)
            break;
    }
    file6.close();
```

```
//====ПУЗЫРЬКОВАЯ СОРТИРОВКА====//
int sortirovka::pyzirek(int N){
    QFile file1("D:\\Rezult Algotim\\test.txt");
    QTime Time;
    Time.start();
    double* digitals = new double[N];
    memset(digitals,1,N);
    srand(time(NULL));
    for(int i=0;i<N;i++) {
        digitals[i]=rand()%1000;
    }
    int start = Time.elapsed();
    for(int i = 1; i < N; ++i)  {
        for(int r = 0; r < N-i; r++)  {
            if(digitals[r] < digitals[r+1])  {
                int temp = digitals[r];
                digitals[r] = digitals[r+1];
                digitals[r+1] = temp;
            }
        }
    }
    int stop = Time.elapsed();
    file1.open(QIODevice::WriteOnly);
    QTextStream writeStream(&file1);
    for (int i = 0; i < N; i++) {
        writeStream <<i<<") "<< digitals[i] << " \n";
    }
    file1.close();
    delete[] digitals;
    int a = stop - start;

    return a;
}
```

```
//======СОРТИРОВКА ВСТАВКАМИ=====//
int sortirovka::vstavka(int N){
    QFile file1("D:\\Rezult Algotim\\test.txt");
    QTime Time;
    Time.start();
    double* digitals = new double[N];
    memset(digitals,1,N);
    srand(time(NULL));
    for(int i=0;i<N;i++){
        digitals[i]=rand()%1000;
    }
    int start = Time.elapsed();
    int key=0;
    int temp=0;
    for (int i=0; i<N-1; i++) {
        key=i+1;
        temp=digitals[key];
        for (int j=i+1; j>0; j--) {
            if (temp<digitals[j-1]){
                digitals[j]=digitals[j-1];
                key=j-1;
            }
        }
        digitals[key]=temp;
    }
    int stop = Time.elapsed();
    file1.open(QIODevice::WriteOnly);
    QTextStream writeStream(&file1);
    for (int i = 0; i < N; i++) {
        writeStream <<i<<") "<< digitals[i] << " \n";
    }
    file1.close();
    delete[] digitals;
    int a = stop - start;
    return a;}
```

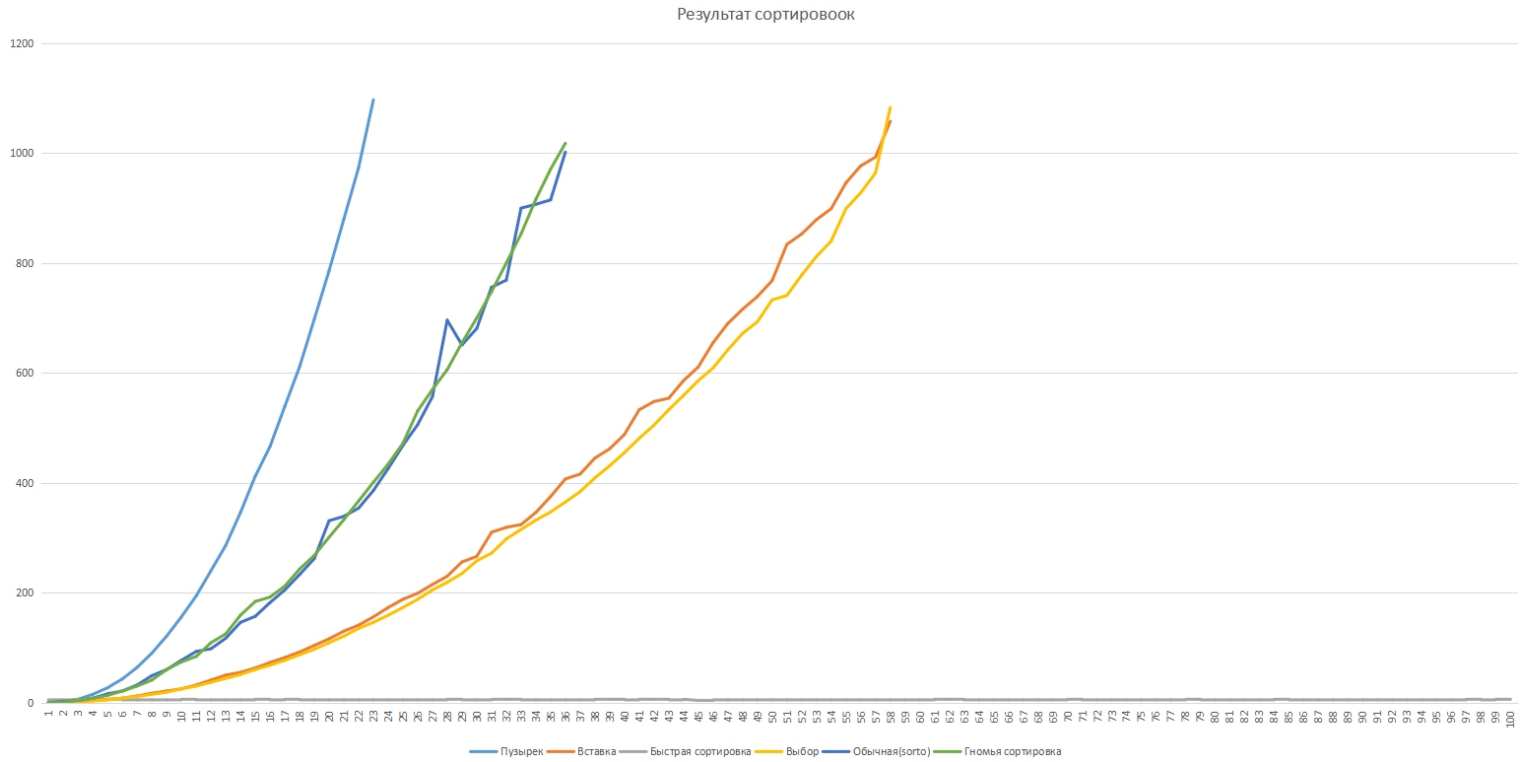
```
//====БЫСТРАЯ СОРТИРОВК==========//
int sortirovka::vstqsortRecursiveavka(int N){
    QFile file1("D:\\Rezult Algotim\\test.txt");
    QTime Time;
    Time.start();
    int* digitals = new int[N];
    memset(digitals,1,N);
    srand(time(NULL));
    for(int i=0;i<N;i++)
        digitals[i]=rand()%1000;
    int start = Time.elapsed();
    qsortRecursive(digitals,N);
    int stop = Time.elapsed();
    file1.open(QIODevice::WriteOnly);
    QTextStream writeStream(&file1);
    for (int i = 0; i < N; i++) {
        writeStream <<i<<") "<< digitals[i] << " \n";
    }
    file1.close();
    delete[] digitals;
    int a = stop - start;
    return a;
}
void sortirovka::qsortRecursive(int* mas, int size) {
    int i = 0;
    int j = size - 1;
    int mid = mas[size / 2];
    do {
        while(mas[i] < mid)
            i++;
        while(mas[j] > mid)
            j--;
        if (i <= j) {
            int tmp = mas[i];
            mas[i] = mas[j];
            mas[j] = tmp;
            i++;
            j--;
        }
    } while (i <= j);
    if(j > 0) {
        qsortRecursive(mas, j + 1);
    }
    if (i < size) {
        qsortRecursive(&mas[i], size - i);
    }
}
```

```
//=======СОРТИРОВКА ВЫБОРОМ======//
int sortirovka::vibor(int N){
    QFile file1("D:\\Rezult Algotim\\test.txt");
    QTime Time;
    Time.start();
    int* digitals = new int[N];
    memset(digitals,1,N);
    srand(time(NULL));
    for(int i=0;i<N;i++)
        digitals[i]=rand()%1000;
    int start = Time.elapsed();
    for (int i = 0; i < N - 1; ++i){
        int smallestIndex = i;
        for (int currentIndex = i + 1; currentIndex < N;
++currentIndex){
            if (digitals[currentIndex] < digitals[smallestIndex])
                smallestIndex = currentIndex;
        }
        swap(digitals[i], digitals[smallestIndex]);
    }
    int stop = Time.elapsed();
    file1.open(QIODevice::WriteOnly);
    QTextStream writeStream(&file1);
    for (int i = 0; i < N; i++) {
        writeStream <<i<<") "<< digitals[i] << " \n";
    }
    file1.close();
    delete[] digitals;
    int a = stop - start;
    stop=0;
    start =0;
    return a;
}
```

```
//======СОРТИРОВКА ПРОСТАЯ======//
int sortirovka::sorto(int N){
    QFile file1("D:\\Rezult Algotim\\test.txt");
    QTime Time;
    Time.start();
    int* digitals = new int[N];
    memset(digitals,1,N);
    srand(time(NULL));
    for(int i=0;i<N;i++)
        digitals[i]=rand()%1000;
    int start = Time.elapsed();

    int tmp;
    for (int i=0;i<N-1;++i) {
        for (int j=i+1; j<N;++j) {
            if (digitals[j]/10<digitals[i]/10) {
                tmp=digitals[i];
                digitals[i]=digitals[j];
                digitals[j]=tmp;
            }
        }
    }

    int stop = Time.elapsed();

    file1.open(QIODevice::WriteOnly);

    QTextStream writeStream(&file1);


    for (int i = 0; i < N; i++) {
        writeStream <<i<<") "<< digitals[i] << " \n";
    }

    file1.close();
    delete[] digitals;
    int a = stop - start;

    return a;
}
```

```
//=======ГНОМЬЯ СОРТИРОВКА======//
int sortirovka::Gnome_sort(int N){
    QFile file1("D:\\Rezult Algotim\\test.txt");
    QTime Time;
    Time.start();
    int* digitals = new int[N];
    memset(digitals,1,N);
    srand(time(NULL));
    for(int i=0;i<N;i++)
        digitals[i]=rand()%1000;
    int start = Time.elapsed();
    int i = 0;
    while (i < N) {
    if (i == 0 || digitals[i-1] <= digitals[i])
        i++;
    else {
        int tmp = digitals[i];
        digitals[i] = digitals[i-1];
        digitals[--i] = tmp;}
    }
    int stop = Time.elapsed();
    file1.open(QIODevice::WriteOnly);
    QTextStream writeStream(&file1);
    for (int i = 0; i < N; i++) {
        writeStream <<i<<") "<< digitals[i] << " \n";
    }
    file1.close();
    delete[] digitals;
    int a = stop - start;
    return a;
}
```

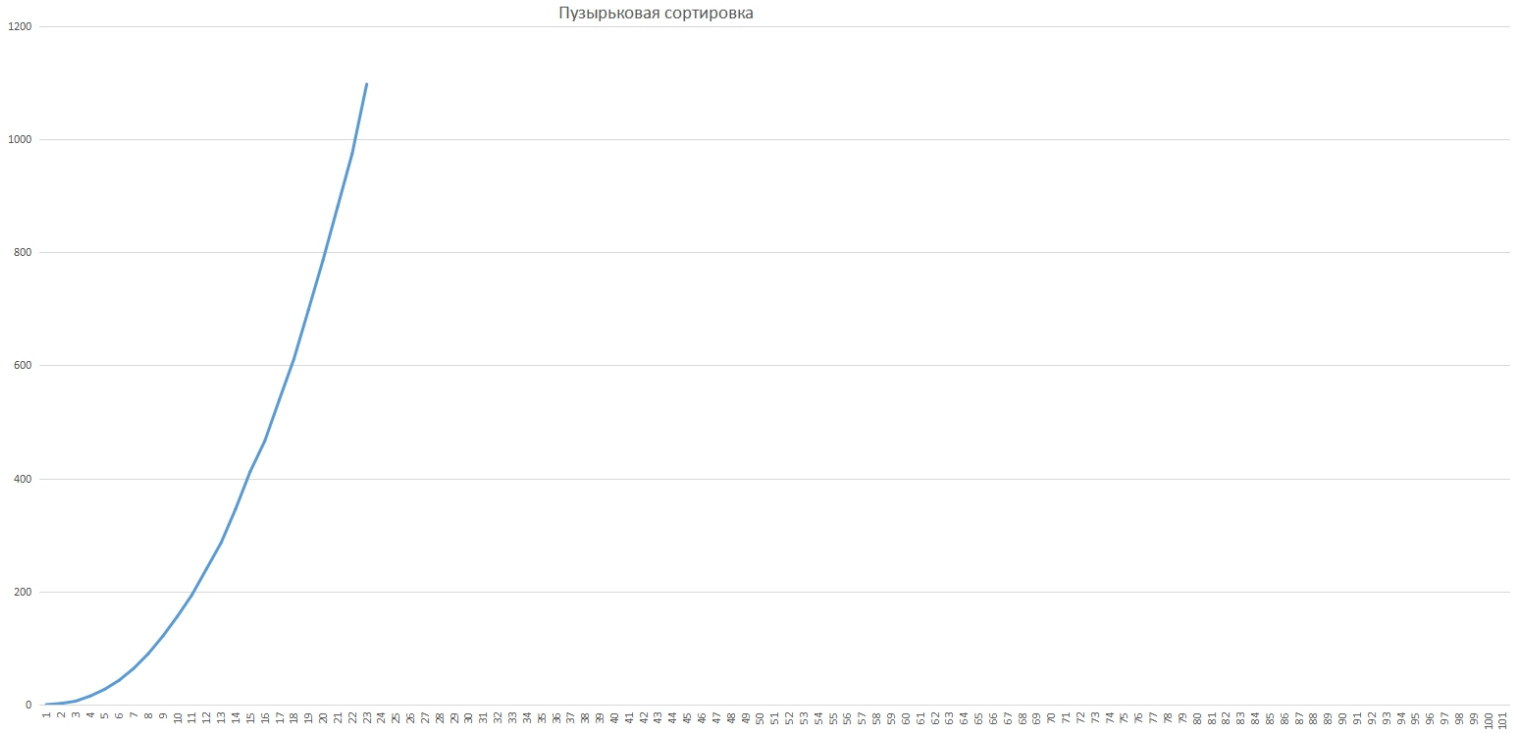
Обработка данных каждой сортировки (построение графиков)



Глава 2

Алгоритмы сортировок, я рассмотрю график каждой сортировки отдельно

Сортировка Пузырьком

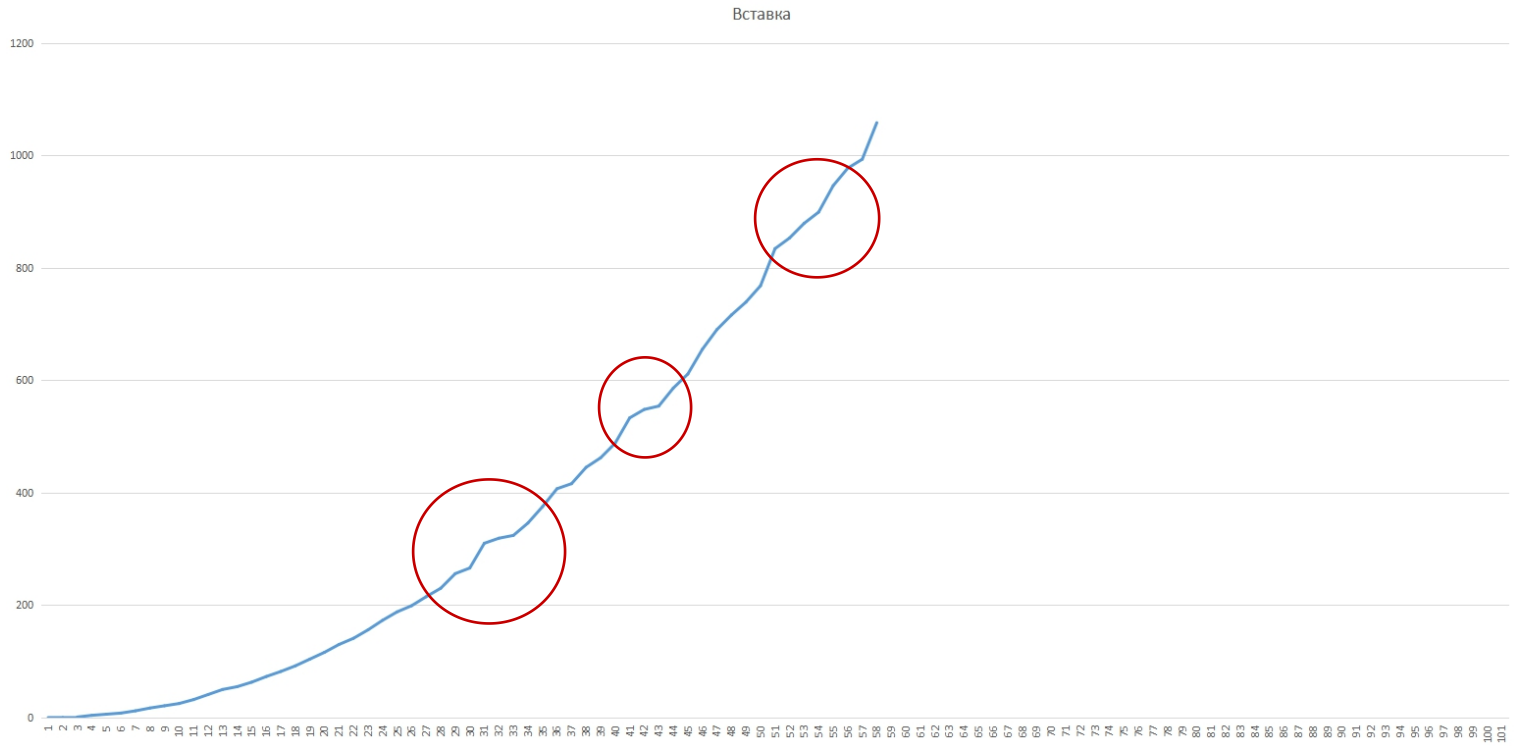


Уже при 23000 элементов в массиве, пузырьковая сортировка сортирует больше 1 секунды

Сортировка вставками

Уже при 57000 элементов в массиве, пузырьковая сортировка сортирует больше 1 секунды.

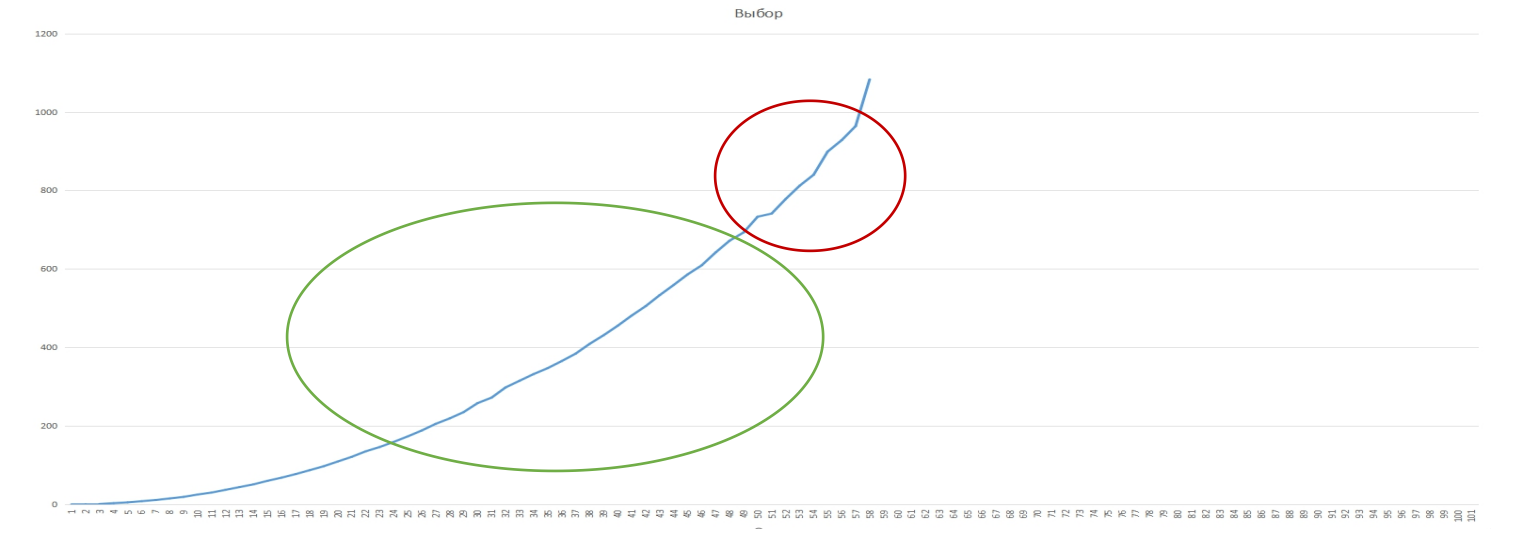
На выделенных красным не плавность графика можно объяснить фоновыми программами



Сортировка быстрая

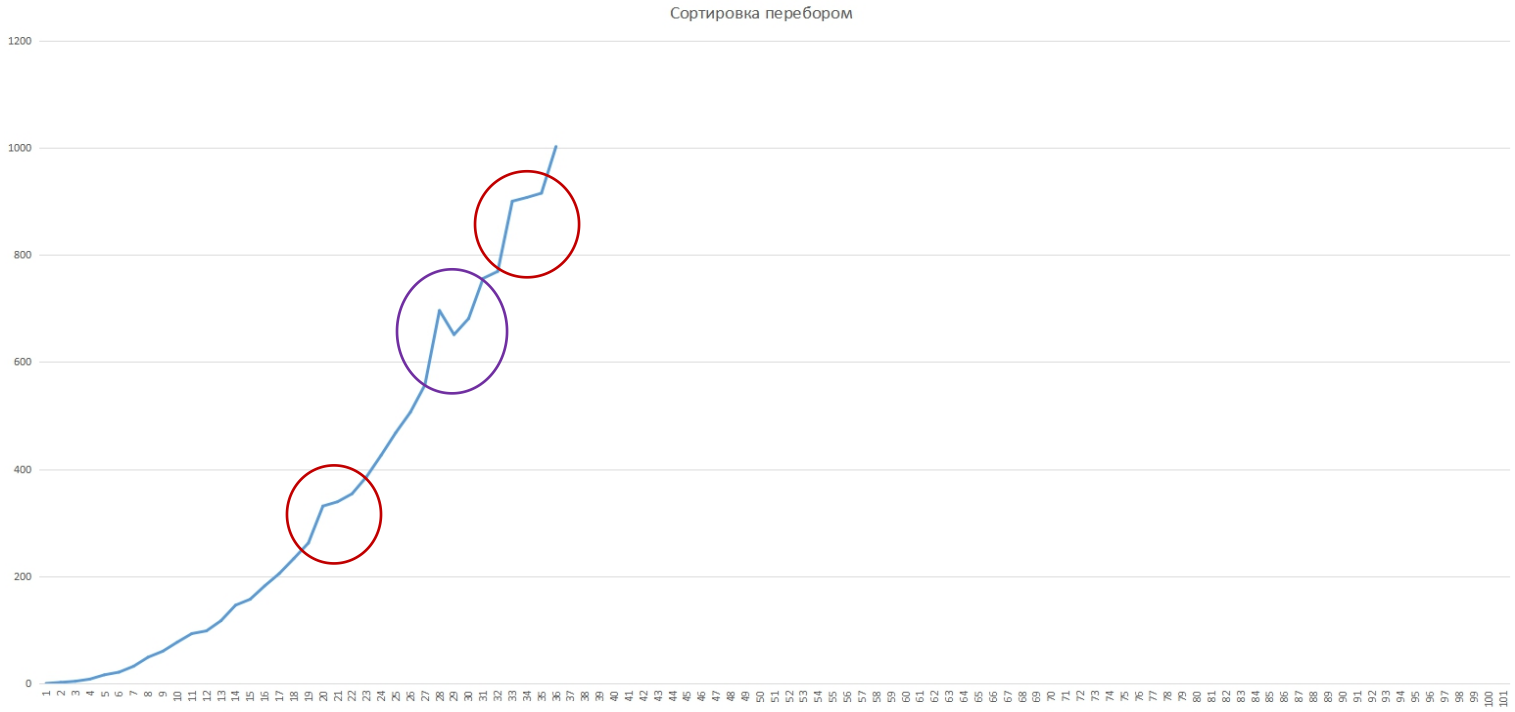
Быстрая сортировка является самой быстрой, в сравнении с остальными. Неровности на графике можно объяснить не удачным рядом чисел.

Сортировка выбором



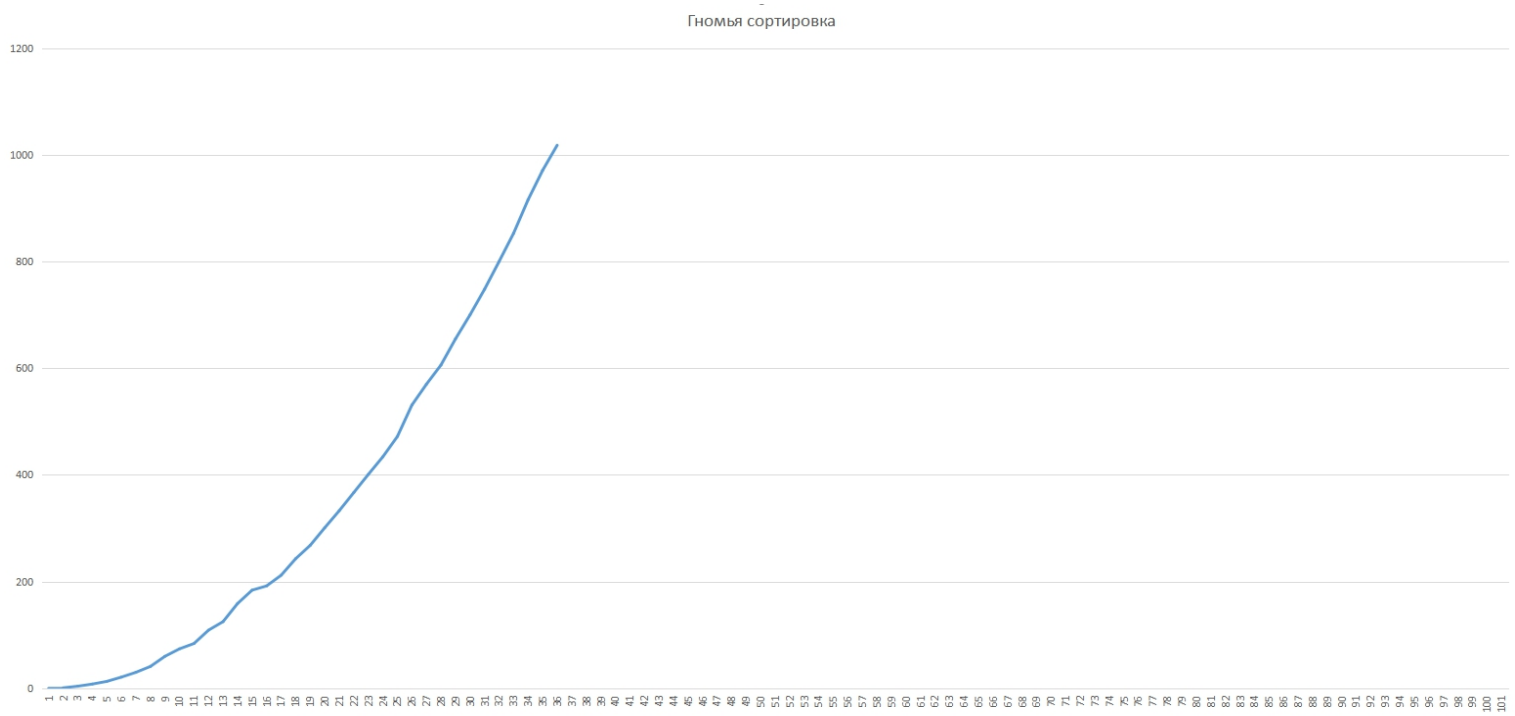
По результатам очень похожа на сортировку вставками, но более плавная . Участок выделенный зелёным сортируется более медленно в отличии от аналогичного участка на сортировке вставками. Не ровность на красном участке объясняется запуском другой программы.

Сортировка перебором



Чуть лучше пузырьковой сортировки. На участках выделенных красным скачки объясняются фоновой активностью других программ. Участок выделенный фиолетовом объясним запуском другой программы.

Гномья сортировка



Гномья сортировка более плавная в сравнении с сортировкой перебором и пузырьком

Глава 3

Вывод

Самой быстрой сортировкой оказалась “Быстрая сортировка” , за ней идут сортировки выбором и вставками. Самой медленной и не эффективной оказалась сортировка пузырьком.