Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Российский химико-технологический университет имени Д.И. Менделеева»
Кафедра информационных компьютерных технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

Выполнил студент группы.....КС-30.....(Бушуев Владимир)
Ссылка на репозиторий: .....(https://github.com/MUCTR-IKT-CPP/VSBushuev_ks30_2023_sem2)

Приняли: .....Пысин Максим Дмитриевич
.....Краснов Дмитрий Олегович
.....Лобанов Алексей Владимирович
.....Крашенинников Роман Сергеевич

Дата сдачи: ..... (15.03.2023)

Оглавление

Описание задачи.

В рамках лабораторной работы необходимо изучить и реализовать метод сортировки вставками.
Для реализованного метода сортировки необходимо провести серию тестов для всех значений N из списка (1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000), при этом:

- в каждом тесте необходимо по 20 раз генерировать вектор, состоящий из N элементов
- каждый элемент массива заполняется случайным числом с плавающей запятой от -1 до 1, для этого можно использовать как C функцию rand(), так и C++ генераторы:
- каждый массив после генерации необходимо отсортировать и замерить время, требуемое на сортировку, для замера времени использовать следующий код:
- Результат замера для каждой попытки каждого теста записать в файл общий файл.
- При замере времени выбираем только один из варивантов (nano_diff, micro_diff, milli_diff, sec_diff).

По окончанию всех тестов необходимо нанести все точки, полученные в результате замеров времени на график где на ось абсцисс(X) нанести N, а на ось ординат(Y) нанести значения времени на сортировку. По полученным точкам построить график лучшего (минимальное время для каждого N), худшего (максимальное время для каждого N) и среднего (среднее время для каждого N) случая.

В качестве дополнительного задания, необходимо построить график худшего случая, и график O(c * g(N)), где g(N)=n^2 соответствует асимптотической сложности рассматриваемого метода сортировки, подобрав такое значение C, что бы начиная с N ~ 1000 график асимптотической сложности возрастал быстрее чем полученное худшее время, но при этом был различим на графике.

Описание метода/модели.

Сортировка вставками — квадратичный алгоритм сортировки.

Алгоритм

Задача заключается в следующем: есть часть массива, которая уже отсортирована, и требуется вставить остальные элементы массива в отсортированную часть, сохранив при этом упорядоченность. Для этого на каждом шаге алгоритма мы выбираем один из элементов входных данных и вставляем его на нужную позицию в уже отсортированной части массива, до тех пор пока весь набор входных данных не будет отсортирован. Метод выбора очередного элемента из исходного массива произволен, однако обычно (и с целью получения устойчивого алгоритма сортировки), элементы вставляются по порядку их появления во входном массиве.

Так как в процессе работы алгоритма могут меняться местами только соседние элементы, каждый обмен уменьшает число инверсий на единицу. Следовательно, количество обменов равно количеству инверсий в исходном массиве вне зависимости от реализации сортировки. Максимальное количество

инверсий содержится в массиве, элементы которого отсортированы по невозрастанию. Число инверсий в таком массиве $n(n-1)/2$.

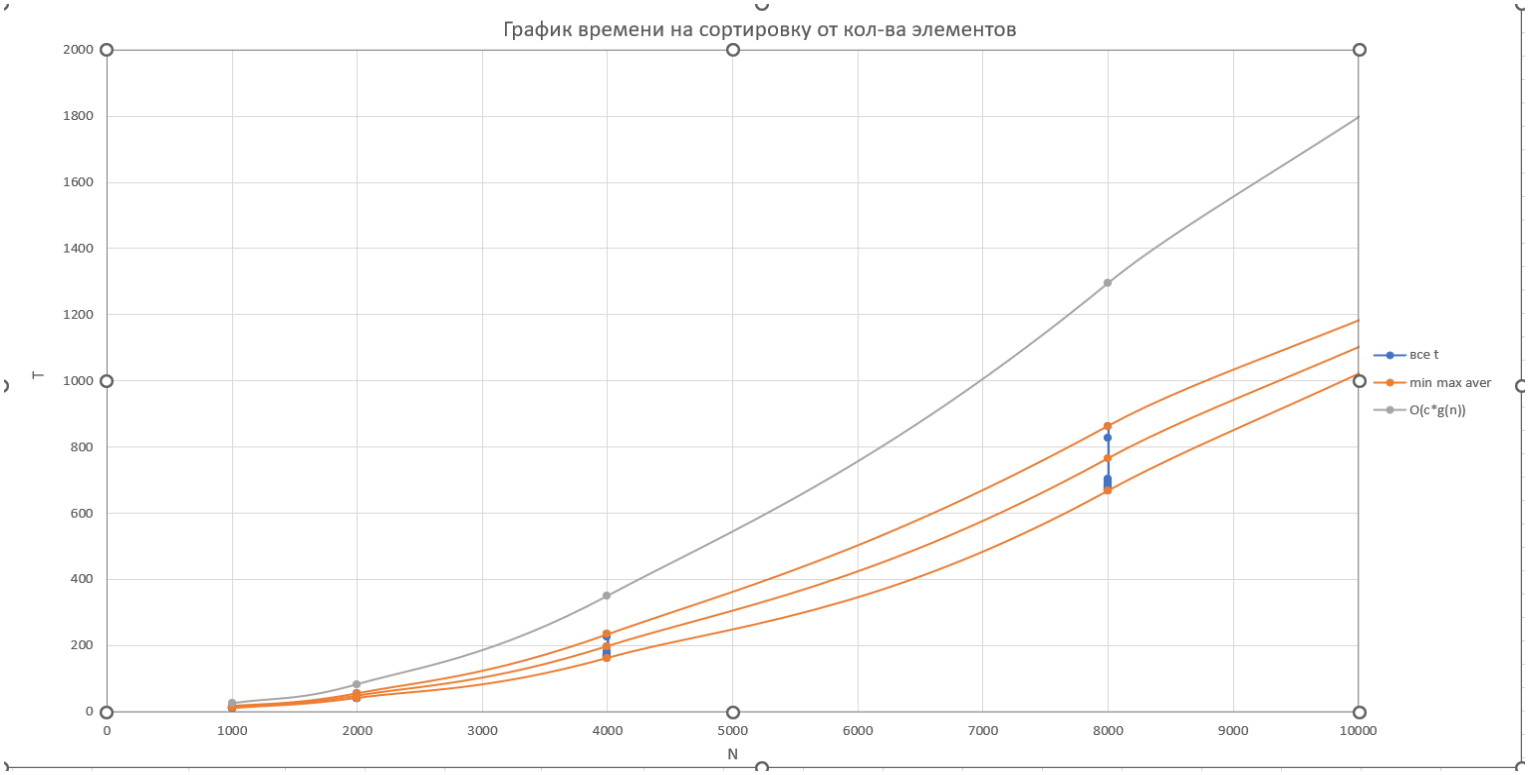
Алгоритм работает за $O(n+k)$, где k— число обменов элементов входного массива, равное числу инверсий. В среднем и в худшем случае — за $O(n^2)$. Минимальные оценки встречаются в случае уже упорядоченной исходной последовательности элементов, наихудшие — когда они расположены в обратном порядке.

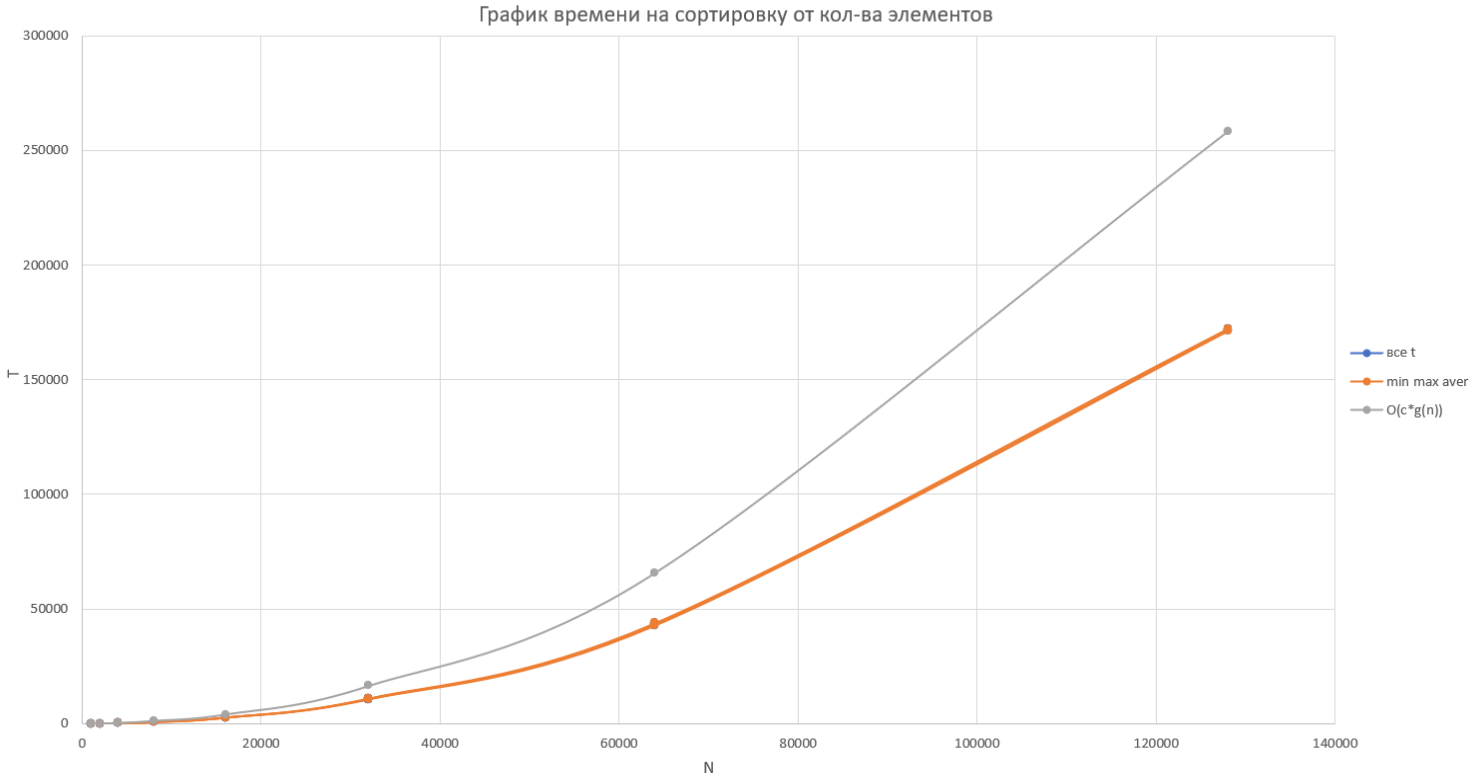
Выполнение задачи.

Язык C++.

```
#include <iostream>
#include <vector>
#include <random>
#include <chrono>
#include <fstream>
#include <string>

using namespace std;

int main()
{
    int i, j;
    double k;
    int n;
    int spis[] = { 1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000};
    const int  e= sizeof(spis) / sizeof(spis[0]);
    double b[e][20];

    ofstream ut("D:\\time.txt", ios::out);
    ut.close();

    fstream out("D:\\time.txt", ios::app);

    for (int s = 0; s < e; s++) {
        n = spis[s];       //отд эл спис
        vector<double> a(n);

        for (int c = 0; c < 20; c++) {  //20 раз генерирую вектор

            mt19937 engine(time(0));
            uniform_real_distribution<double> gen(-1.0, 1.0);
            for (auto& el : a)
                el = gen(engine);   //генератор случ чисел

            chrono::high_resolution_clock::time_point start = chrono::high_resolution_clock::now();
            for (i = 1; i < n; i++) {  //сортировка вставками. Бегу по n эл, j для
                j = i - 1;
                while ((j >= 0)) {
                    while (a[j] > a[j + 1]) {
                        k = a[j];
                        a[j] = a[j + 1];
                        a[j + 1] = k;
                    }
                    j = j - 1;
                }


            }
            chrono::high_resolution_clock::time_point end = chrono::high_resolution_clock::now();
            chrono::duration<double, milli> milli_diff = end - start;

            b[s][c] = milli_diff.count();  //двумерный массив всех значений 8 20

            string  ss =to_string(milli_diff.count());
```

```
        ss.replace(ss.find("."),1, ",");


        out << n << "\t" << ss << endl;
    }
    out << endl;
}
out.close();

double mini[e];
double maxi[e];
double aver[e];
double cgn[e];

ofstream mm("D:\\mm.txt", ios::out);
mm.close();

fstream minim("D:\\mm.txt", ios::app);

for (int s = 0; s < e; s++) {
    double *min = min_element(std::begin(b[s]), std::end(b[s]));
    double *max = max_element(std::begin(b[s]), std::end(b[s]));
    mini[s] = *min;
    maxi[s] = *max;
    aver[s] =(maxi[s]+mini[s]) / 2;
    cgn[s] = (1.5) * maxi[s];


}
for (int s = 0; s < e; s++) {
    n = spis[s];
    minim <<n<<"\t" << mini[s]<<endl;

}
minim << endl;
for (int s = 0; s < e; s++) {
    n = spis[s];
    minim << n << "\t" << aver[s] << endl;

}
minim << endl;
for (int s = 0; s < e; s++) {
    n = spis[s];
    minim << n << "\t" << maxi[s] << endl;

}
minim.close();

ofstream cgfil("D:\\cgn.txt", ios::out);
cgfil.close();

fstream cg("D:\\cgn.txt", ios::app);
for (int s = 0; s < e; s++) {
    n = spis[s];
    cg << n << "\t" << cgn[s] << endl;
}
cg.close();

  }
```





Заключение.

Метод сортировки вставками медленный. Потребовалось времени 1 час 16 мин.