Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Реферат

По дисциплине «Основы алгоритмизации и программирования»

На тему «Сравнение скорости работы основных сортировок (+ возможно новая)»

Выполнила:

Студент(ка) 1 курса 7 группы

Подшиваленко Диана Игоревна

Проверил:

Белодед Николай Иванович

2023, Минск

**Содержание**

1. Предисловие … 3
2. Реализация сортировок … 4
3. Тестирование … 7
4. Результаты … 8

**Предисловие**

В данном реферате будут рассмотрены такие основные сортировки: сортировка пузырьком, сортировка выбором, сортировка вставками, быстрая сортировка. (и возможно новая сортировка)

**Реализация сортировок**

#include <iostream>

#include<Windows.h>

#include <conio.h>

#include <random>

#include <ctime>

using namespace std;

void bubble\_sort(int\*& b, int n) { // пузырьковая сортировка

while (n--) {

bool t = false;

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

if (b[i] > b[i + 1]) {

swap(b[i], b[i + 1]);

t = true;

}

}

if (!t) {

break;

}

}

}

void insert\_sort(int\*& c, int n) { // сортировка вставками

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

int sort = i - 1;

while (sort > -1 and c[sort] > c[sort + 1]) {

swap(c[sort], c[sort + 1]);

sort--;

}

}

}

void choice\_sort(int\*& d, int n) { // сортировка выбором

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

int min\_poz = i;

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

if (d[j] < d[min\_poz]) {

min\_poz = j;

}

}

if (min\_poz != i) {

swap(d[i], d[min\_poz]);

}

}

}

void quick\_sort(int\*& e, int first, int last) { // быстрая сортировка

int middle;

int l = first, r = last;

middle = e[(l + r) / 2];

while (l < r) {

while (e[l] < middle) l++;

while (e[r] > middle) r--;

if (l <= r) {

swap(e[l], e[r]);

l++;

r--;

}

}

if (first < r) quick\_sort(e, first, r);

if (l < last) quick\_sort(e, l, last);

}

void mysort(int\*& a, int n) { // моя сортировка

int i = 0;

while (i <= n - i - 1) {

int min = i;

int max = n - i - 1;

for (int j = i; j < n - i; j++) {

if (a[j] < a[min]) {

min = j;

}

if (a[j] > a[max]) {

max = j;

}

}

if (i == max and n - i - 1 == min) {

swap(a[i], a[n - i - 1]);

}

else if (i == max) {

int temp = a[n - i - 1];

a[n - i - 1] = a[max];

a[i] = a[min];

a[min] = temp;

}

else if (n - i - 1 == min) {

int temp = a[i];

a[i] = a[min];

a[n - i - 1] = a[max];

a[max] = temp;

}

else {

swap(a[i], a[min]);

swap(a[n - i - 1], a[max]);

}

i++;

}

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

srand(time(NULL));

int n;

cin >> n;

int\* a = new int[n];

int\* b = new int[n];

int\* c = new int[n];

int\* d = new int[n];

int\* e = new int[n];

int\* f = new int[n];

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

a[i] = b[i] = c[i] = d[i] = e[i] = f[i] = rand() % 100000 + 1;

}

cout << "Исходный массив:\n";

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

cout << a[i] << ' ';

}

cout << '\n';

clock\_t start = clock();

bubble\_sort(b, n);

clock\_t end = clock();

cout << "Пузырьковая сортировка:\n";

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

cout << b[i] << ' ';

}

cout << '\n';

double duration = (float)(end - start) / CLOCKS\_PER\_SEC;

printf("Время работы = %.5f секунд\n", duration);

start = clock();

insert\_sort(c, n);

end = clock();

cout << "Сортировка вставками:\n";

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

cout << c[i] << ' ';

}

cout << '\n';

duration = (float)(end - start) / CLOCKS\_PER\_SEC;

printf("Время работы = %.5f секунд\n", duration);

start = clock();

choice\_sort(d, n);

end = clock();

cout << "Сортировка выбором:\n";

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

cout << d[i] << ' ';

}

cout << '\n';

double duration = (float)(end - start) / CLOCKS\_PER\_SEC;

printf("Время работы = %.5f секунд\n", duration);

start = clock();

quick\_sort(e, 0, n - 1);

end = clock();

cout << "Быстрая сортировка:\n";

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

cout << e[i] << ' ';

}

cout << '\n';

duration = (float)(end - start) / CLOCKS\_PER\_SEC;

printf("Время работы = %.5f секунд\n", duration);

start = clock();

mysort(f, n);

end = clock();

cout << "Моя сортировка:\n";

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

cout << f[i] << ' ';

}

cout << '\n';

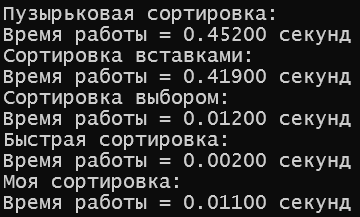
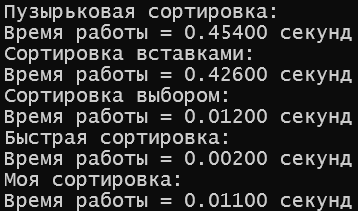
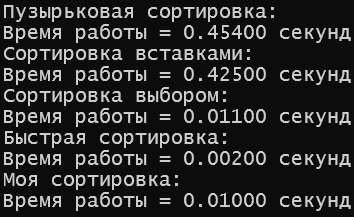
duration = (float)(end - start) / CLOCKS\_PER\_SEC;

printf("Время работы = %.5f секунд\n", duration);

}

**Тестирование**

Будем тестировать сортировки на разном количестве чисел. На каждое будем отводить 3 попытки для более точного результата и высчитывать среднее время работы.

1) 5000 чисел

Среднее время:

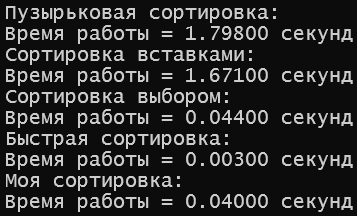
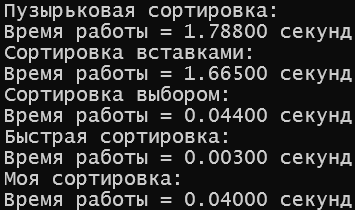
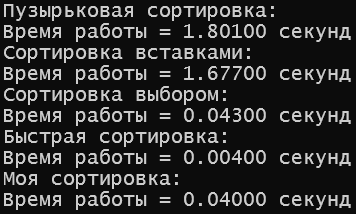
Пузырьковая сортировка: 0.45333 секунд

Сортировка вставками: 0.42333 секунд

Сортировка выбором: 0.01167 секунд

Быстрая сортировка: 0.002 секунд

Моя сортировка: 0.01067 секунд

2) 10000 чисел

Среднее время:

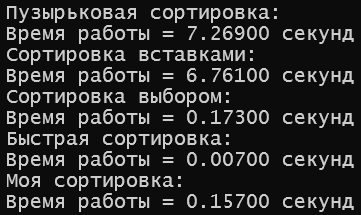
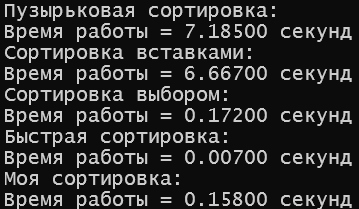
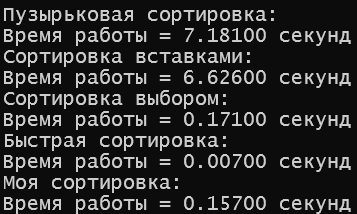
Пузырьковая сортировка: 1.79567 секунд

Сортировка вставками: 1.67100 секунд

Сортировка выбором: 0.04367 секунд

Быстрая сортировка: 0.00333 секунд

Моя сортировка: 0.04000 секунд

3) 20000 чисел

Среднее время:

Пузырьковая сортировка: 7.21167 секунд

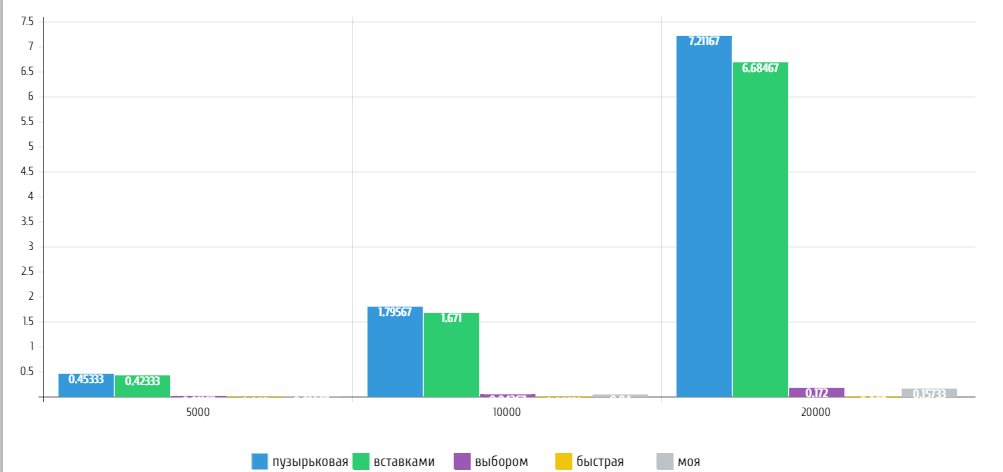
Сортировка вставками: 6.68467 секунд

Сортировка выбором: 0.17200 секунд

Быстрая сортировка: 0.00700 секунд

Моя сортировка: 0.15733 секунд

**Результаты**

На основании полученных результатов построим график

Как видно, наилучшее время показала быстрая сортировка. На 2 месте моя сортировка, на третьем – сортировка выбором.