**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по практической работе №2**

**по дисциплине «Программирование»**

Тема: ОДНОМЕРНЫЕ СТАТИЧЕСКИЕ МАССИВЫ

|  |  |  |
| --- | --- | --- |
| Студент(ка) гр. | Чекстер А. А. |  |
| Преподаватель | Глущенко А. Г. |  |

Санкт-Петербург

2022

**Цель работы.**

Изучение одномерных статических массивов. Изучение алгоритмов сортировки и поиска.

**Основные теоретические положения.**

Массив представляет собой индексированную последовательность однотипных элементов с заранее определенным количеством элементов. Наглядно одномерный массив можно представить, как набор пронумерованных ячеек, в каждой из которых содержится определенное значение.

Элементы массива нумеруются с нуля. При описании массива используются те же модификаторы (класс памяти, const и инициализатор), что и для простых переменных.

Сортировка – процесс размещения элементов заданного множества объектов в определенном порядке. Когда элементы отсортированы, их проще найти, производить с ними различные операции. Сортировка напрямую влияет на скорость алгоритма, в котором нужно обратиться к определенному элементу массива.

Простейшая из сортировок – сортировка обменом (пузырьковая сортировка). Вся суть метода заключается в попарном сравнении элементов и последующем обмене. Таким образом, если следующий элемент меньше текущего, то они меняются местами, максимальный элемент массива постепенно смещается в конец массива, а минимальный – в начало. Один полный проход по массиву может гарантировать, что в конце массива находится максимальный элемент.

Быстрая сортировка (quick sort) – одна из самых быстрых сортировок. Эта сортировка по сути является существенно улучшенной версией алгоритма пузырьковой сортировки.

Общая идея алгоритма состоит в том, что сначала выбирается из массива элемент, который называется опорным. От выбора опорного элемента не зависит корректность алгоритма, но в отдельных случаях может сильно зависеть его эффективность. Затем необходимо сравнить все остальные элементы с опорным и переставить их в массиве так, чтобы разбить массив на три непрерывных отрезка, следующие друг за другом: меньше опорного, раны опорному и больше опорного. Для меньших и больших значений необходимо выполнить рекурсивно ту же последовательность операций, если длина отрезка больше единицы.

Алгоритм бинарного поиска – классический алгоритм поиска в отсортированном массиве, который использует дробление массива на половины. Если элемент, который необходимо найти, присутствует в списке, то бинарный поиск возвращает ту позицию, в которой он был найден.

**Постановка задачи.**

Необходимо написать программу, которая:

1)    Создает целочисленный массив размерности *N* =100. Элементы массивы должны принимать случайное значение в диапазоне от -99 до 99.

2)    Отсортировать заданный в пункте 1 массив […] сортировкой (от меньшего к большему). Определить время, затраченное на сортировку, используя библиотеку chrono.

3)    Найти максимальный и минимальный элемент массива. Подсчитайте время поиска этих элементов в отсортированном массиве и неотсортированном, используя библиотеку chrono.

4)    Выводит среднее значение (если необходимо, число нужно округлить) максимального и минимального значения. Выводит индексы всех элементов, которые равны этому значению, и их количество.

5)    Выводит количество элементов в отсортированном массиве, которые меньше числа *a*, которое инициализируется пользователем.

6)    Выводит количество элементов в отсортированном массиве, которые больше числа *b*, которое инициализируется пользователем.

7)    Выводит информацию о том, есть ли введенное пользователем число в отсортированном массиве. Реализуйте алгоритм бинарного поиска. Сравните скорость его работы с обычным перебором. (\*)

8)     Меняет местами элементы массива, индексы которых вводит пользователь. Выведите скорость обмена, используя библиотеку chrono.

Должна присутствовать возможность запуска каждого пункта многократно.

**Выполнение работы.**

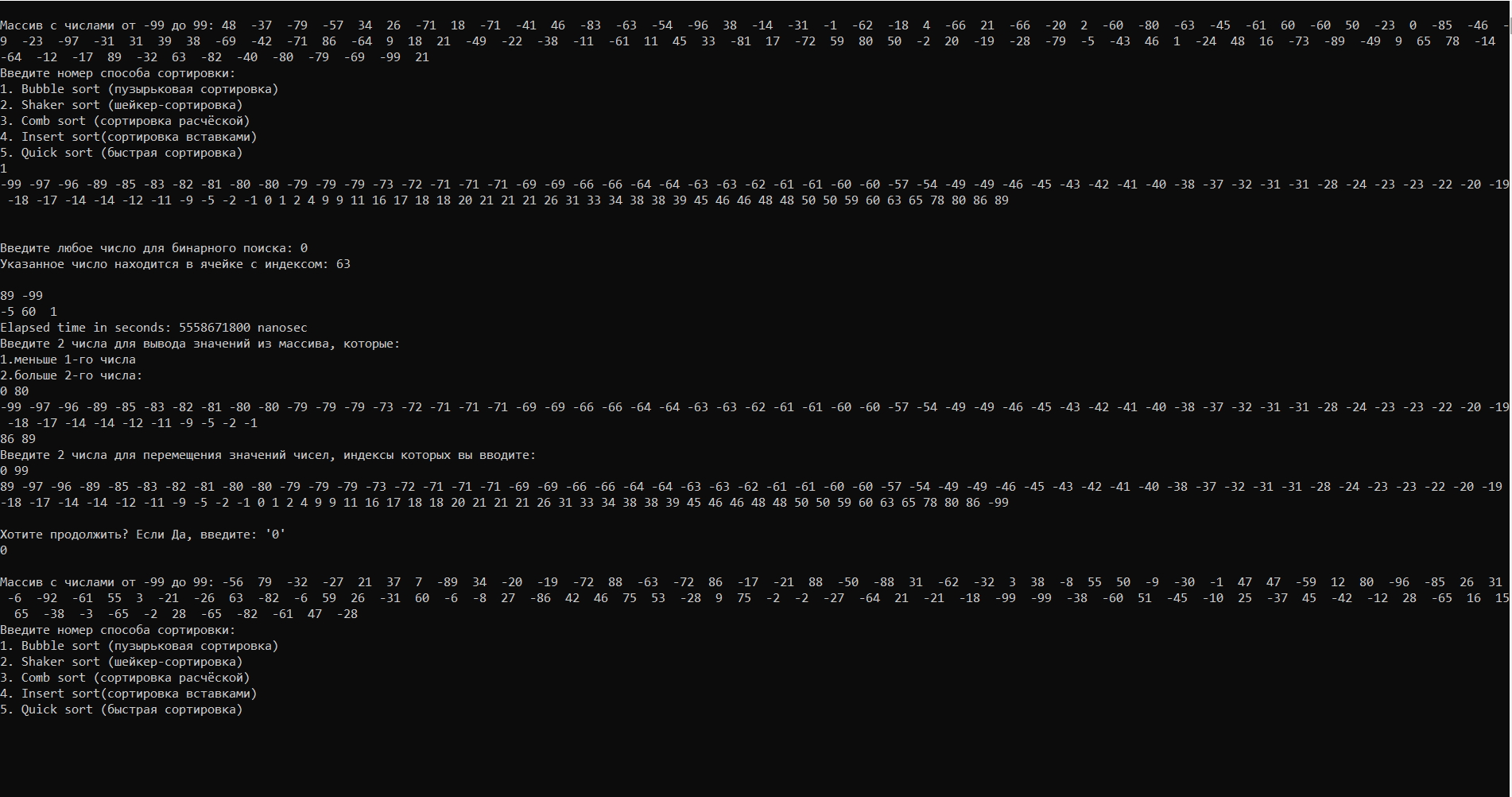
Код программы представлен в приложении А.

Блок описания кода и использованных алгоритмов

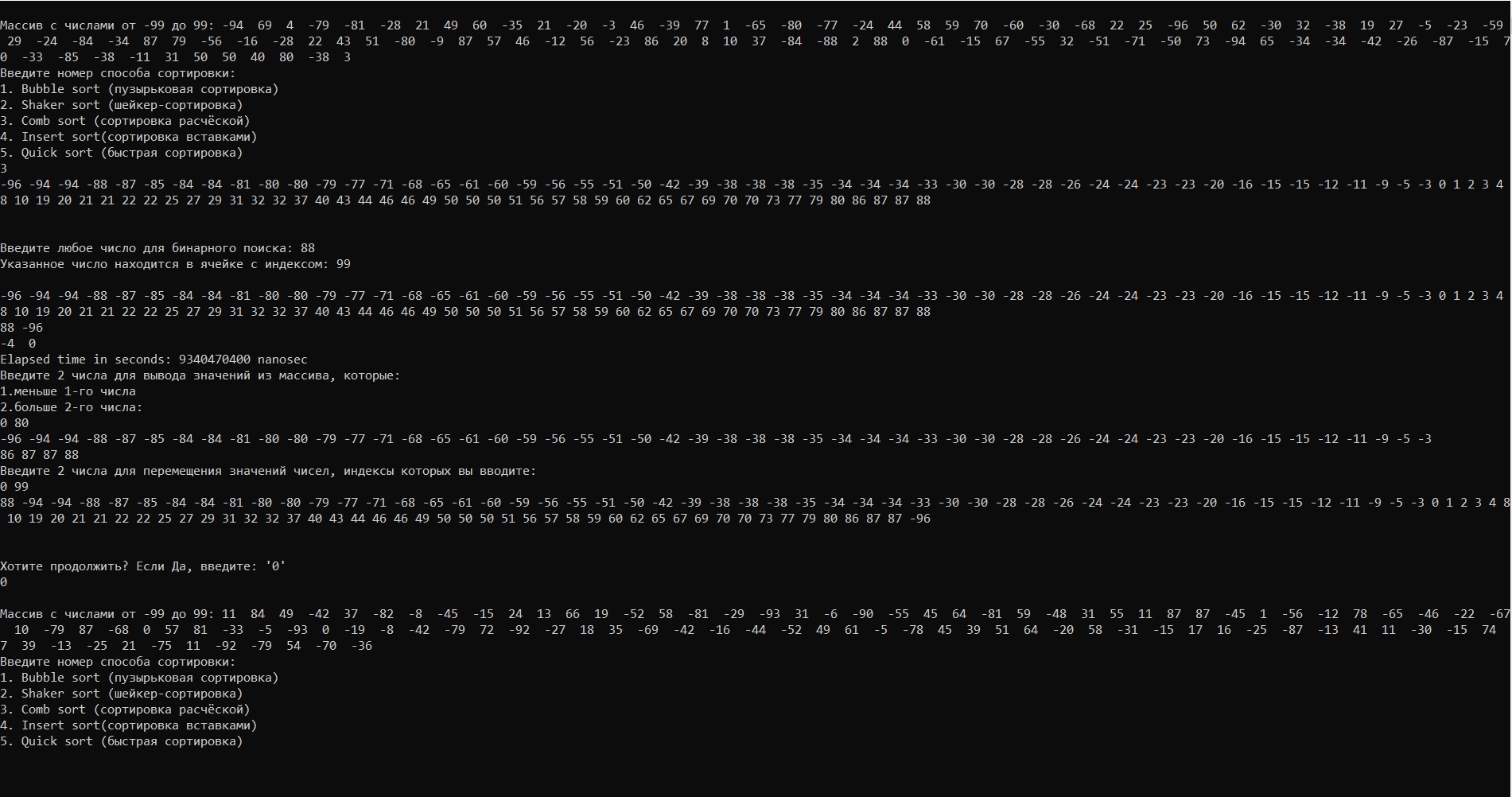
Для выполнения второго пункта реализована возможность выбора сортировки для сравнения времени работы каждого алгоритма. Для выбора опорного элемента написан алгоритм подсчёта среднего значения и поиска среди значений массива, ближайшего к среднему. Для измерения времени выполнения той или иной функции используются объекты типа steady\_clock из библиотеки chrono. Для выполнения третьего пункта реализован алгоритм поиска наибольшего и наименьшего значений перебором. Для выполнения четвёртого пункта я просто сортирую массив и работаю с первым и последним значениями, т. к. они являются минимальным и максимальным соответственно. В пятом и шестом пунктах просто перебором считаем количество элементов больше или меньше введённого значения. В седьмом пункте реализован алгоритм бинарного пункта без рекурсии. В восьмом пункте просто считываем два значения и меняем их местами как в любой сортировке.

**Блок скриншотов работы программы**

Скриншот 1



Скриншот 2



**Выводы.**

Изучен навык работы с одномерными статическими массивами. Изучены базовые алгоритмы сортировок и поиска. Изучен навык бинарного поиска.

Бинарный поиск может быть медленнее на таких множествах значений как у нас (малого размера), но при увеличении размера массива бинарный поиск будет показывать куда большую скорость, чем обычный перебор.

Приложение А

рабочий код

#include <iostream>

#include <cstdlib>

#include <ctime>

#include <chrono>

using namespace std;

using namespace chrono;

//функция для сортировки - 5

void quicksort(int\* randArr, int end, int begin)

{

//auto start = chrono::steady\_clock::now();

int mid,swap;

int f = begin;

int l = end;

mid = randArr[(f + l) / 2];

while (f < l)

{

while (randArr[f] < mid) f++;

while (randArr[l] > mid) l--;

if (f <= l)

{

swap = randArr[f];

randArr[f] = randArr[l];

randArr[l] = swap;

f++;

l--;

}

}

if (begin < l) quicksort(randArr, l, begin);

if (f < end) quicksort(randArr, end, f);

/\*auto end = chrono::steady\_clock::now();

cout << "Elapsed time in seconds: " << chrono::duration\_cast<chrono::nanoseconds>(end - start).count() << " nanosec" << endl;\*/

}

int Binarny(int arr[], int left, int right, int key)

{

int midd = 0;

while (1)

{

midd = (left + right) / 2;

if (key < arr[midd]) // если искомое меньше значения в ячейке

right = midd - 1; // смещаем правую границу поиска

else if (key > arr[midd]) // если искомое больше значения в ячейке

left = midd + 1; // смещаем левую границу поиска

else // иначе (значения равны)

return midd; // функция возвращает индекс ячейки

if (left > right) // если границы сомкнулись

return -1;

}

}

int main()

{

int snachala;

while (true) {

setlocale(LC\_ALL, "rus");

int randArr[100]{};

int first, second;

srand(time(NULL));

for (int i = 0; i < 100; i++)

{

randArr[i] = rand() % 189 - 99; // -99 ... 99

}

cout << endl << "Массив c числами oт -99 до 99: ";

for (int i = 0; i < 100; i++)

{

cout << randArr[i] << " ";

}

cout << endl;

int sort\_num;

cout << "Введите номер способа сортировки: " << endl;

cout << "1. Bubble sort (пузырьковая сортировка)" << endl;

cout << "2. Shaker sort (шейкер-сортировка)" << endl;

cout << "3. Comb sort (сортировка расчёской)" << endl;

cout << "4. Insert sort(сортировка вставками)" << endl;

cout << "5. Quick sort (быстрая сортировка)" << endl;

cin >> sort\_num;

// 1 - Bubble sort (пузырьковая сортировка)

if (sort\_num == 1) {

int key = 0;

int index = 0;

auto start = chrono::steady\_clock::now();

int const n = 100;

int k = 0;

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

for (k = 0; k < n - j - 1; ++k)

if (randArr[k] > randArr[k + 1]) {

double x = randArr[k];

randArr[k] = randArr[k + 1];

randArr[k + 1] = x;

}

}

for (int o = 0; o < n; o++)

cout << randArr[o] << ' ';

cout << endl;

cout << "\n\nВведите любое число для бинарного поиска: ";

cin >> key;

index = Binarny(randArr, 0, n, key);

if (index >= 0)

cout << "Указанное число находится в ячейке с индексом: " << index << "\n\n";

else

cout << "В массиве нет такого числа!\n\n";

cout << randArr[99] << ' ' << randArr[0] << endl;

float middle\_num;

middle\_num = (randArr[0] + randArr[99]) / 2;

if (int(middle\_num \* 100) % 100 > 50)

middle\_num = int(middle\_num) + 1;

else

middle\_num = int(middle\_num);

cout << middle\_num << " ";

int count\_i = 0;

for (int d = 0; d < 100; d++) {

if (randArr[d] == middle\_num) {

cout << d << " ";

count\_i += 1;

}

}

cout << " ";

cout << count\_i << endl;

auto end = chrono::steady\_clock::now();

cout << "Elapsed time in seconds: " << chrono::duration\_cast<chrono::nanoseconds>(end - start).count() << " nanosec" << endl;

int a, b;

cout << "Введите 2 числа для вывода значений из массива, которые: " << endl;

cout << "1.меньше 1-го числа " << endl;

cout << "2.больше 2-го числа: " << endl;

cin >> a >> b;

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

if (randArr[l] < a)

cout << randArr[l] << ' ';

}

cout << endl;

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

if (randArr[p] > b)

cout << randArr[p] << ' ';

}

cout << endl;

int q, w, t, z;

cout << "Введите 2 числа для перемещения значений чисел, индексы которых вы вводите: " << endl;

cin >> q >> w;

z = randArr[w];

t = randArr[q];

for (int u = 0; u < 100; ++u) {

if (u == q) {

randArr[q] = z;

//cout << z << " ";

}

if (u == w) {

randArr[w] = t;

cout << t << " ";

}

else

cout << randArr[u] << " ";

}

}

// 2 - Shaker sort (шейкер-сортировка)

if (sort\_num == 2) {

int key = 0;

int index = 0;

auto start = chrono::steady\_clock::now();

int const n = 100;

int c = 0;

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

for (int k = 0; k < n - j - 1; ++k)

if (randArr[k] > randArr[k + 1]) {

double x = randArr[k];

randArr[k] = randArr[k + 1];

randArr[k + 1] = x;

c += 1;

}

}

if (c >= n - 1) {

for (int j = n - 1; j > 0; --j) {

for (int k = n - 1; k > 1; --k)

if (randArr[k] > randArr[k + 1]) {

double x = randArr[k];

randArr[k] = randArr[k + 1];

randArr[k + 1] = x;

c += 1;

}

}

}

for (int o = 0; o < n; ++o)

if (o != 2)

cout << randArr[o] << ' ';

cout << endl;

cout << "\n\nВведите любое число для бинарного поиска: ";

cin >> key;

index = Binarny(randArr, 0, n, key);

if (index >= 0)

cout << "Указанное число находится в ячейке с индексом: " << index << "\n\n";

else

cout << "В массиве нет такого числа!\n\n";

cout << randArr[99] << ' ' << randArr[0] << endl;

float middle\_num;

middle\_num = (randArr[0] + randArr[99]) / 2;

if (int(middle\_num \* 100) % 100 > 50)

middle\_num = int(middle\_num) + 1;

else

middle\_num = int(middle\_num);

cout << middle\_num << " ";

int count\_i = 0;

for (int d = 0; d < 100; d++) {

if (randArr[d] == middle\_num) {

cout << d << " ";

count\_i += 1;

}

}

cout << " ";

cout << count\_i << endl;

auto end = chrono::steady\_clock::now();

cout << "Elapsed time in seconds: " << chrono::duration\_cast<chrono::nanoseconds>(end - start).count() << " nanosec" << endl;

int a, b;

cout << "Введите 2 числа для вывода значений из массива, которые: " << endl;

cout << "1.меньше 1-го числа " << endl;

cout << "2.больше 2-го числа: " << endl;

cin >> a >> b;

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

if (randArr[l] < a)

cout << randArr[l] << ' ';

}

cout << endl;

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

if (randArr[p] > b)

cout << randArr[p] << ' ';

}

cout << endl;

int q, w, t, z;

cout << "Введите 2 числа для перемещения значений чисел, индексы которых вы вводите: " << endl;

cin >> q >> w;

z = randArr[w];

t = randArr[q];

for (int u = 0; u < 100; ++u) {

if (u == q) {

randArr[q] = z;

//cout << z << " ";

}

if (u == w) {

randArr[w] = t;

cout << t << " ";

}

else

cout << randArr[u] << " ";

}

}

// 3 - Comb sort (сортировка расчёской)

if (sort\_num == 3) {

int key = 0;

int index = 0;

auto start = chrono::steady\_clock::now();

int swap;

int const n = 100;

float k = 1.247, S = n - 1;

int count = 0;

while (S >= 1)

{

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

{

if (randArr[i] > randArr[int(i + S)])

{

swap = randArr[int(i + S)];

randArr[int(i + S)] = randArr[i];

randArr[i] = swap;

}

}

S /= k;

}

while (true)

{

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

{

if (randArr[i] > randArr[i + 1])

{

swap = randArr[i + 1];

randArr[i + 1] = randArr[i];

randArr[i] = swap;

}

else count++;

}

if (count == n - 1)

break;

else

count = 0;

}

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

{

cout << randArr[i] << " ";

}

cout << endl;

cout << "\n\nВведите любое число для бинарного поиска: ";

cin >> key;

index = Binarny(randArr, 0, n, key);

if (index >= 0)

cout << "Указанное число находится в ячейке с индексом: " << index << "\n\n";

else

cout << "В массиве нет такого числа!\n\n";

for (int o = 0; o < n; o++)

cout << randArr[o] << ' ';

cout << endl;

cout << randArr[99] << ' ' << randArr[0] << endl;

float middle\_num;

middle\_num = (randArr[0] + randArr[99]) / 2;

if (int(middle\_num \* 100) % 100 > 50)

middle\_num = int(middle\_num) + 1;

else

middle\_num = int(middle\_num);

cout << middle\_num << " ";

int count\_i = 0;

for (int d = 0; d < 100; d++) {

if (randArr[d] == middle\_num) {

cout << d << " ";

count\_i += 1;

}

}

cout << " ";

cout << count\_i << endl;

auto end = chrono::steady\_clock::now();

cout << "Elapsed time in seconds: " << chrono::duration\_cast<chrono::nanoseconds>(end - start).count() << " nanosec" << endl;

int a, b;

cout << "Введите 2 числа для вывода значений из массива, которые: " << endl;

cout << "1.меньше 1-го числа " << endl;

cout << "2.больше 2-го числа: " << endl;

cin >> a >> b;

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

if (randArr[l] < a)

cout << randArr[l] << ' ';

}

cout << endl;

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

if (randArr[p] > b)

cout << randArr[p] << ' ';

}

cout << endl;

int q, w, t, z;

cout << "Введите 2 числа для перемещения значений чисел, индексы которых вы вводите: " << endl;

cin >> q >> w;

z = randArr[w];

t = randArr[q];

for (int u = 0; u < 100; ++u) {

if (u == q) {

randArr[q] = z;

}

if (u == w) {

randArr[w] = t;

cout << t << " ";

}

else

cout << randArr[u] << " ";

}

cout << endl;

}

// 4 - Insert sort (сортировка вставками)

if (sort\_num == 4) {

int key = 0;

int index = 0;

auto start = chrono::steady\_clock::now();

int const n = 100;

int k = 0, c = 0;

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

c = randArr[j];

for (k = j - 1; k >= 0 && randArr[k] > c; k--)

randArr[k + 1] = randArr[k];

randArr[k + 1] = c;

}

auto end = chrono::steady\_clock::now();

cout << "Elapsed time in seconds: " << chrono::duration\_cast <chrono::nanoseconds>(end - start).count() << " nanosec" << endl;

for (int o = 0; o < n; o++)

cout << randArr[o] << " ";

cout << endl;

cout << "\n\nВведите любое число для бинарного поиска: ";

cin >> key;

index = Binarny(randArr, 0, n, key);

if (index >= 0)

cout << "Указанное число находится в ячейке с индексом: " << index << "\n\n";

else

cout << "В массиве нет такого числа!\n\n";

cout << randArr[0] << ' ' << randArr[99] << endl;

float middle\_num;

middle\_num = (randArr[0] + randArr[99]) / 2;

if (int(middle\_num \* 100) % 100 > 50)

middle\_num = int(middle\_num) + 1;

else

middle\_num = int(middle\_num);

cout << middle\_num << " ";

int count\_i = 0;

for (int d = 0; d < 100; d++) {

if (randArr[d] == middle\_num) {

cout << d << " ";

count\_i += 1;

}

}

cout << " ";

cout << count\_i << endl;

int a, b;

cout << "Введите 2 числа для вывода значений из массива, которые: " << endl;

cout << "1.меньше 1-го числа " << endl;

cout << "2.больше 2-го числа: " << endl;

cin >> a >> b;

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

if (randArr[l] < a)

cout << randArr[l] << ' ';

}

cout << endl;

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

if (randArr[p] > b)

cout << randArr[p] << ' ';

}

cout << endl;

int q, w, t, z;

cout << "Введите 2 числа для перемещения значений чисел, индексы которых вы вводите: " << endl;

cin >> q >> w;

z = randArr[w];

t = randArr[q];

for (int u = 0; u < 100; ++u) {

if (u == q) {

randArr[q] = z;

}

if (u == w) {

randArr[w] = t;

cout << t << " ";

}

else

cout << randArr[u] << " ";

}

cout << endl;

return 0;

}

// 5 - Quick sort (быстрая сортировка)

if (sort\_num == 5) {

int key = 0;

int index = 0;

const int n = 100;

int end = n - 1, begin = 0;

quicksort(randArr, end, begin);

cout << endl << "Отсортированный массив: ";

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

cout << randArr[i] << " ";

for (int o = 0; o < n; o++)

cout << randArr[o] << ' ';

cout << endl;

cout << "\n\nВведите любое число для бинарного поиска: ";

cin >> key;

index = Binarny(randArr, 0, n, key);

if (index >= 0)

cout << "Указанное число находится в ячейке с индексом: " << index << "\n\n";

else

cout << "В массиве нет такого числа!\n\n";

cout << randArr[99] << ' ' << randArr[0] << endl;

float middle\_num;

middle\_num = (randArr[0] + randArr[99]) / 2;

if (int(middle\_num \* 100) % 100 > 50)

middle\_num = int(middle\_num) + 1;

else

middle\_num = int(middle\_num);

cout << middle\_num << " ";

int count\_i = 0;

for (int d = 0; d < 100; d++) {

if (randArr[d] == middle\_num) {

cout << d << " ";

count\_i += 1;

}

}

cout << " ";

cout << count\_i << endl;

int a, b;

cout << "Введите 2 числа для вывода значений из массива, которые: " << endl;

cout << "1.меньше 1-го числа " << endl;

cout << "2.больше 2-го числа: " << endl;

cin >> a >> b;

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

if (randArr[l] < a)

cout << randArr[l] << ' ';

}

cout << endl;

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

if (randArr[p] > b)

cout << randArr[p] << ' ';

}

cout << endl;

int q, w, t, z;

cout << "Введите 2 числа для перемещения значений чисел, индексы которых вы вводите: " << endl;

cin >> q >> w;

z = randArr[w];

t = randArr[q];

for (int u = 0; u < 100; ++u) {

if (u == q) {

randArr[q] = z;

}

if (u == w) {

randArr[w] = t;

cout << t << " ";

}

else

cout << randArr[u] << " ";

}

cout << endl;

}

cout << "\n\nХотите продолжить? Если Да, введите: '0'" << endl;

cin >> snachala;

if (snachala != 0)

break;

}

}