Федеральное государственное автономное образовательное учреждение

высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

КАФЕДРА СИСТЕМНОГО АНАЛИЗА И ТЕЛЕКОММУНИКАЦИИ

**ПРОВЕРИЛ ВЫПОЛНИЛ**

Ассистент кафедры САиТ Студент группы КТбо1-2

Старший преподаватель кафедры САиТ Лапшин В.С. Мельничук И.В.

«\_\_» \_\_\_\_\_\_\_\_\_\_2023 г. «\_\_» \_\_\_\_\_\_\_\_\_\_2023 г.

**ЛАБОРАТОРНАЯ РАБОТА №1**

По дисциплине

«Основы алгоритмизации и программирования»

Вариант 6

На тему

«Алгоритмы сортировки»

**Таганрог 2023**

**Цель**: ознакомиться с сортировкой данных и ее видами, а также запрограммировать на языке Си некоторые из алгоритмов сортировок в соответствии с индивидуальным заданием.

**Задачи**: Произвести сортировку алгоритмами: сортировка пузырьком, сортировка перемешиванием, быстрая сортировка

**Ход работы**:

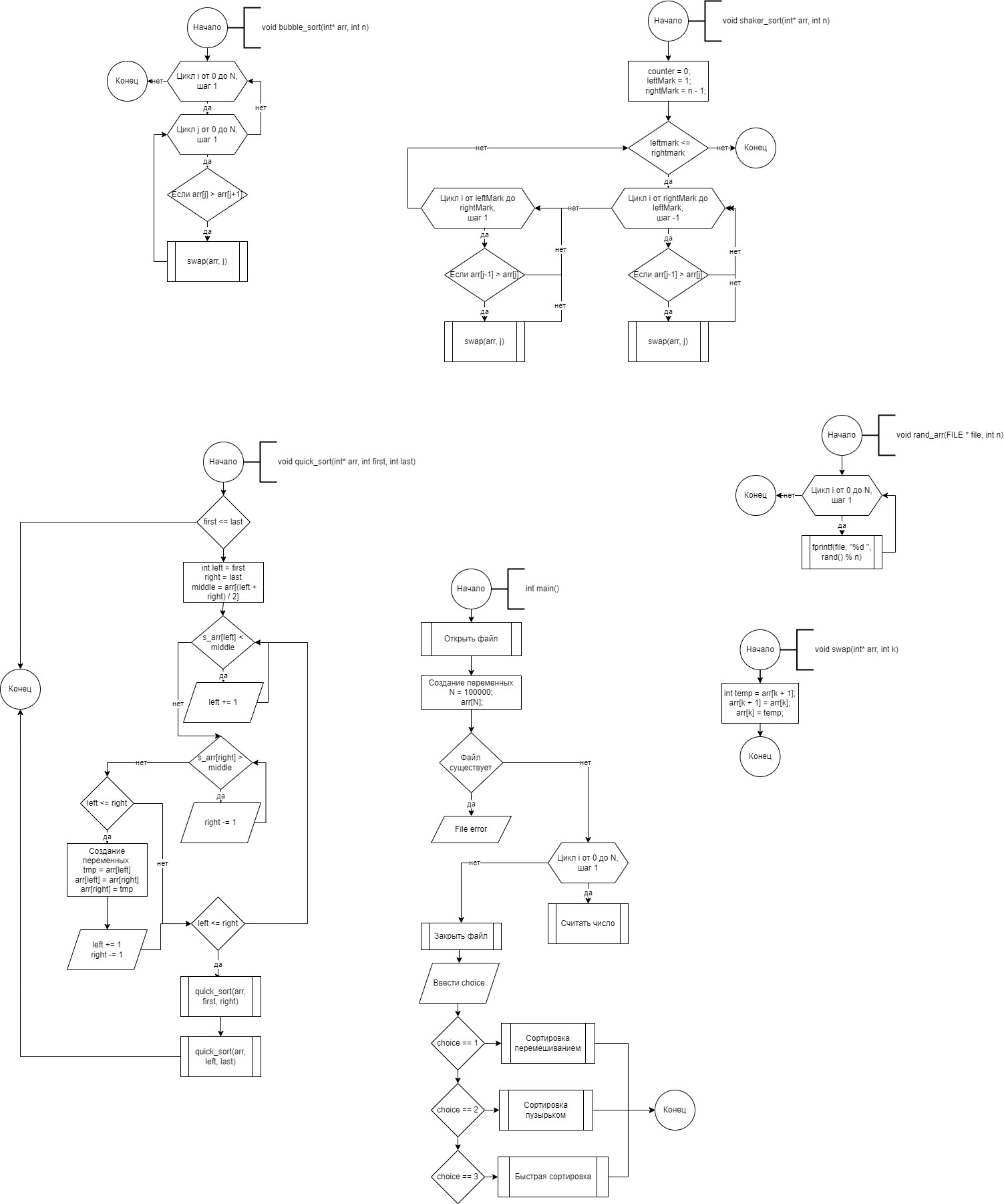
**Составление математической модели**:

Сортировка пузырьком: алгоритм состоит из повторяющихся проходов по сортируемому массиву. За каждый проход элементы последовательно сравниваются попарно и, если порядок в паре неверный, выполняется перестановка элементов. Проходы по массиву повторяются {\displaystyle N-1}N - 1 раз или до тех пор, пока на очередном проходе не окажется, что обмены больше не нужны, что означает — массив отсортирован.

Быстрая сортировка: выбрать элемент из массива. Назовём его опорным. перераспределение элементов в массиве таким образом, что элементы, меньшие опорного, помещаются перед ним, а большие или равные - после. Рекурсивно применить первые два шага к двум подмассивам слева и справа от опорного элемента. Рекурсия не применяется к массиву, в котором только один элемент или отсутствуют элементы.

Сортировка перемешиванием: границы рабочей части массива (то есть части массива, где происходит движение) устанавливаются в месте последнего обмена на каждой итерации. Массив просматривается поочередно справа налево и слева направо.

**Проектирование.**



**Реализация.**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

void swap(int\* arr, int k) {

int temp = arr[k + 1];

arr[k + 1] = arr[k];

arr[k] = temp;

}

void bubble\_sort(int\* arr, int n) {

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

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

if (arr[j] > arr[j + 1]) {

swap(arr, j);

}

}

}

}

void quick\_sort(int\* arr, int first, int last)

{

if (first < last)

{

int left = first, right = last, middle = arr[(left + right) / 2];

do

{

while (arr[left] < middle) left++;

while (arr[right] > middle) right--;

if (left <= right)

{

int tmp = arr[left];

arr[left] = arr[right];

arr[right] = tmp;

left++;

right--;

}

} while (left <= right);

quick\_sort(arr, first, right);

quick\_sort(arr, left, last);

}

}

void rand\_arr(FILE \* file, int n) {

for (int i = 0; i < n; i++) forint(file, "%d ", rand() % n);

}

void shaker\_sort(int\* arr, int n) {

int counter = 0;

int leftMark = 1;

int rightMark = n - 1;

while (leftMark <= rightMark)

{

for (int i = rightMark; i >= leftMark; i--) {

if (arr[i - 1] > arr[i]) {

swap(arr, i - 1);

}

}

leftMark++;

for (int i = leftMark; i <= rightMark; i++) {

if (arr[i - 1] > arr[i]) {

swap(arr, i - 1);

}

counter++;

}

rightMark--;

}

}

int main()

{

FILE \* file = fopen("input.txt", "r");

long long int counter = 0;

const int N = 100000;

int choice;

int arr[N];

if (file == NULL) {

puts("File error");

return 0;

}

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

fscanf(file, "%d", &arr[i]);

};

fclose(file);

std::cout << "Choice sorting algorithm: \n";

std::cout << "1. Quick sort \n";

std::cout << "2. Bubble sort: \n";

std::cout << "3. Shaker sort: \n";

std::cin >> choice;

switch (choice) {

case 1: quick\_sort(arr, 0, N - 1); break;

case 2: bubble\_sort(arr, N); break;

case 3: shaker\_sort(arr, N); break;

default : quick\_sort(arr, 0, N - 1); break;

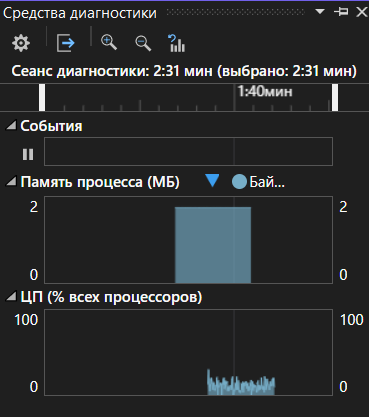
}

}

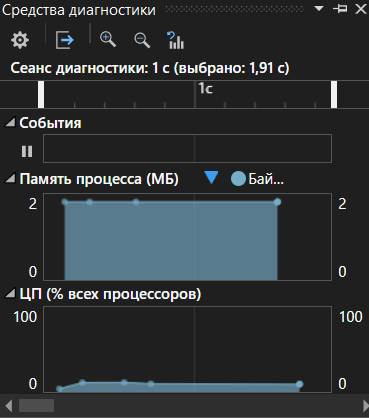
**Результат работы программы.**

|  |  |  |  |
| --- | --- | --- | --- |
| Критерии | Сортировка пузырьком | Быстрая сортировка | Сортировка перемешиванием |
| Время работы | 2:38 мин | 00:01 сек | 1:18 мин |
| Оперативная память | 2 МБ | 2 МБ | 2 МБ |

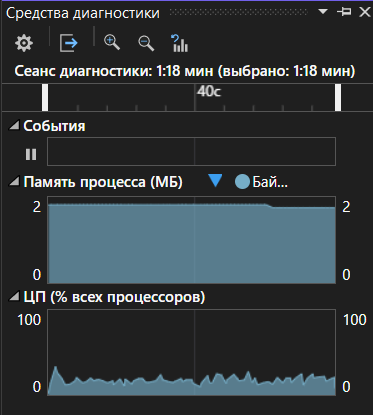
**Сортировка пузырьком**

****

**Быстрая сортировка**

****

**Сортировка перемешиванием**

****

**Вывод:**

В ходе работы были получены знания о структурах различных алгоритмов сортировки, их устройстве, параметрах оценивания, свойства этих алгоритмов, а также их сфере применения.

Наиболее детально были изучены алгоритмы сортировки пузырьком, быстрая и перемешиванием; были написаны программы сортировки для каждого алгоритма и проверены сортировкой набора данных из 100000 чисел.

Лучший результат показала сортировка быстрая сортировка. Следующая по эффективности – сортировка перемешиванием. Наименее эффективной оказалась пузырьковая сортировка.