Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет безопасности информационных технологий

Дисциплина:

«Алгоритмы и структуры данных»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

«Сортировка слиянием»

Выполнила:

Дынина Е.А., студент группы N3249

_____
(подпись)

Проверил:

Ерофеев С.А.

_____
(отметка о выполнении)

_____
(подпись)

Санкт-Петербург

2025 г.

СОДЕРЖАНИЕ

ПОСТАНОВКА ЗАДАЧИ

Цель работы — разработать программу сортировки нисходящим слиянием, в которой числа считываются из файла в статистический или динамический массив по выбору пользователя.

Для выполнения цели поставлены следующие задачи:

- Изучить сортировку слиянием;
- Создать блок-схему алгоритма с обработкой ошибок;
- Написать программу на языке C++ с интерфейсом для пользователей.

ТЕХНИЧЕСКАЯ ЗАДАЧА

Сортировка слиянием заключается в переборе срезов исходного массива размером i с шагом 2i, где i от 1 до n, где n — размер массива. Берутся по два соседних среза, поэлементно сравниваются, и в исходный массив сначала добавляются меньшие значения, а потом, если сравнивать больше нечего, то все остальные. Таким образом, перебрав все срезы, весь массив отсортируется.

Входные данные

На вход в программу через консольное приложение пользователь подает символ choice типа string.

Далее через указанный заранее файл программа принимает массив с числами типа double (диапазон $[-1{,}7 \cdot 10^{308}, 1{,}7 \cdot 10^{308}]$).

Промежуточные данные

В ходе работы программы задействованы промежуточные переменные, указанные в таблице 1.

Таблица 1 – Промежуточные данные

Название переменной	Тип в Python	Диапазон типа	Значение
$file\_path$	string	—	Путь к файлу с числами
$tempnum$	long double	$[-1{,}2 \cdot 10^{4932}, 1{,}2 \cdot 10^{4932}]$	Переменная, в которую считываем числа из файла в первый раз, чтобы проверить входят ли они в диапазон типа double
$size$	int	$[-2147483648, 2147483647]$	Размер массива
cur	int	$[-2147483648, 2147483647]$	Позиция указателя после считывания чисел из файла
$value$	double	$[-1{,}7 \cdot 10^{308}, 1{,}7 \cdot 10^{308}])$	Переменная для заполнения массива числами из файла
$it1$	int	$[-2147483648, 2147483647]$	Индекс для отслеживания текущего положения в левой части среза

$it2$	int	$[-2147483648, 2147483647]$	Индекс для отслеживания текущего положения в правой части среза
i	int	$[-2147483648, 2147483647]$	Переменная для реализации перебора в цикле
j	int	$[-2147483648, 2147483647]$	Переменная для реализации перебора в цикле

Выходные данные

В результате работы программы в консольное приложение выводится отсортированный массив с числами типа double (диапазон $[-1{,}7 \cdot 10^{308}, 1{,}7 \cdot 10^{308}]$).

Используемые функции

- **ifstream** —открывает файл для чтения;
- **.is_open()** — проверяет открыт ли файл;
- **numeric_limits <тип>::lowest** — возвращает наименьшее отрицательное конечное значение типа;
- **numeric_limits <тип>::max** — возвращает максимальное конечное значение типа;
- **.clear()** — сбрасывает состояние потока ifstream;
- **.tellg()** — возвращает текущую позицию указателя в файле;
- **.seekg()** — устанавливает указатель на определенную позицию в файле;
- **new double []** — пытается выделить и инициализировать объект или массив объектов указанного или заполнителя и возвращает подходящий типизированный ненулевой указатель на объект (или начальный объект массива);
- **delete[]** — отменяет выделение блока памяти;
- **.close()** — закрытие файла
- **print (int n, double *nums)** — выводит n элементов массива nums;
- **input (string file_path, int size, double *array)** — вводит из файла file_name size элементов типа double в массив array:
- **merge(double *array, int left, int mid, int right)** — сравнивает два среза массива array, добавляет меньшие значения в массив, а потом добавляет оставшиеся;
- **sorting(int n, double *array)** — определяет срезы и проводит сортировку.

БЛОК-СХЕМА

Рассмотрим блок-схему придуманного алгоритма, где реализована сортировка слиянием и учтены всевозможные ошибки.

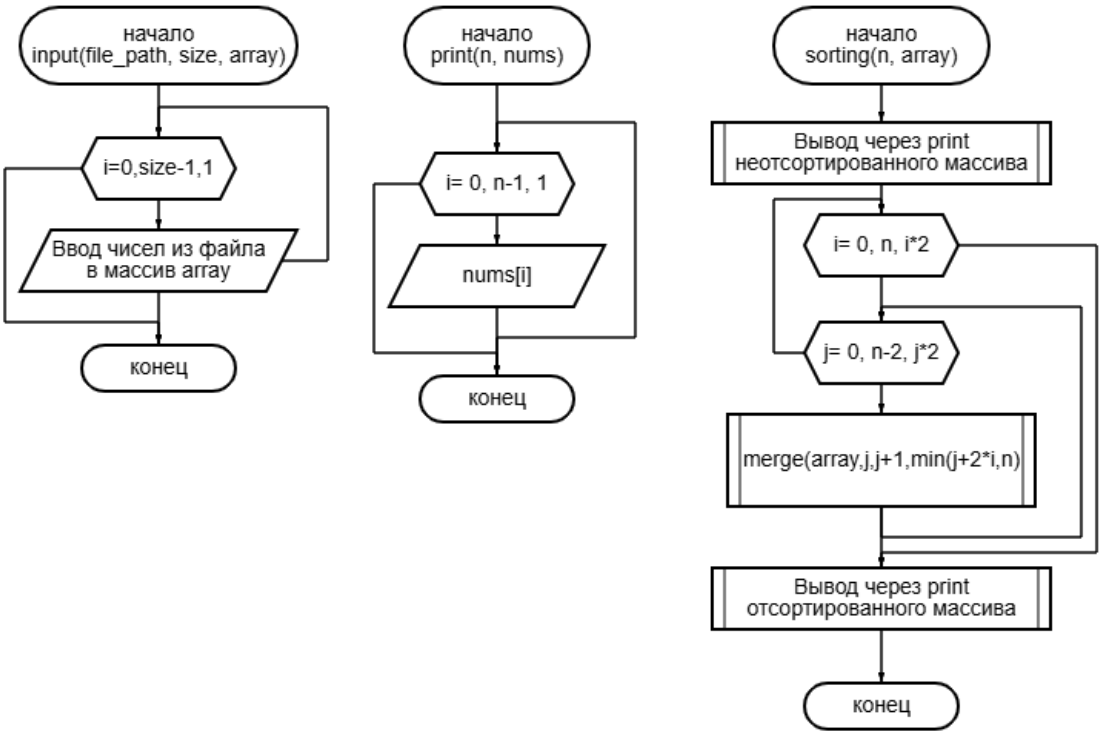

Рисунок 1 – Блок-схема функций ввода, вывода и сортировки

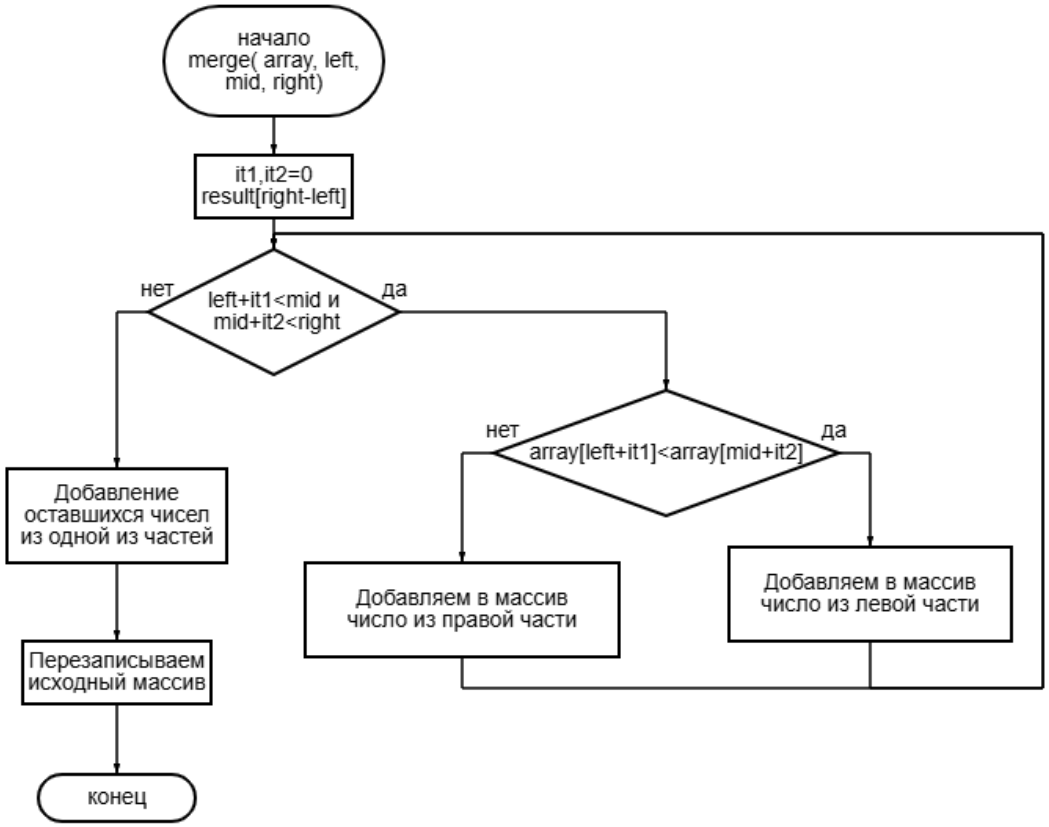

Рисунок 2 – Блок-схема функции слияния

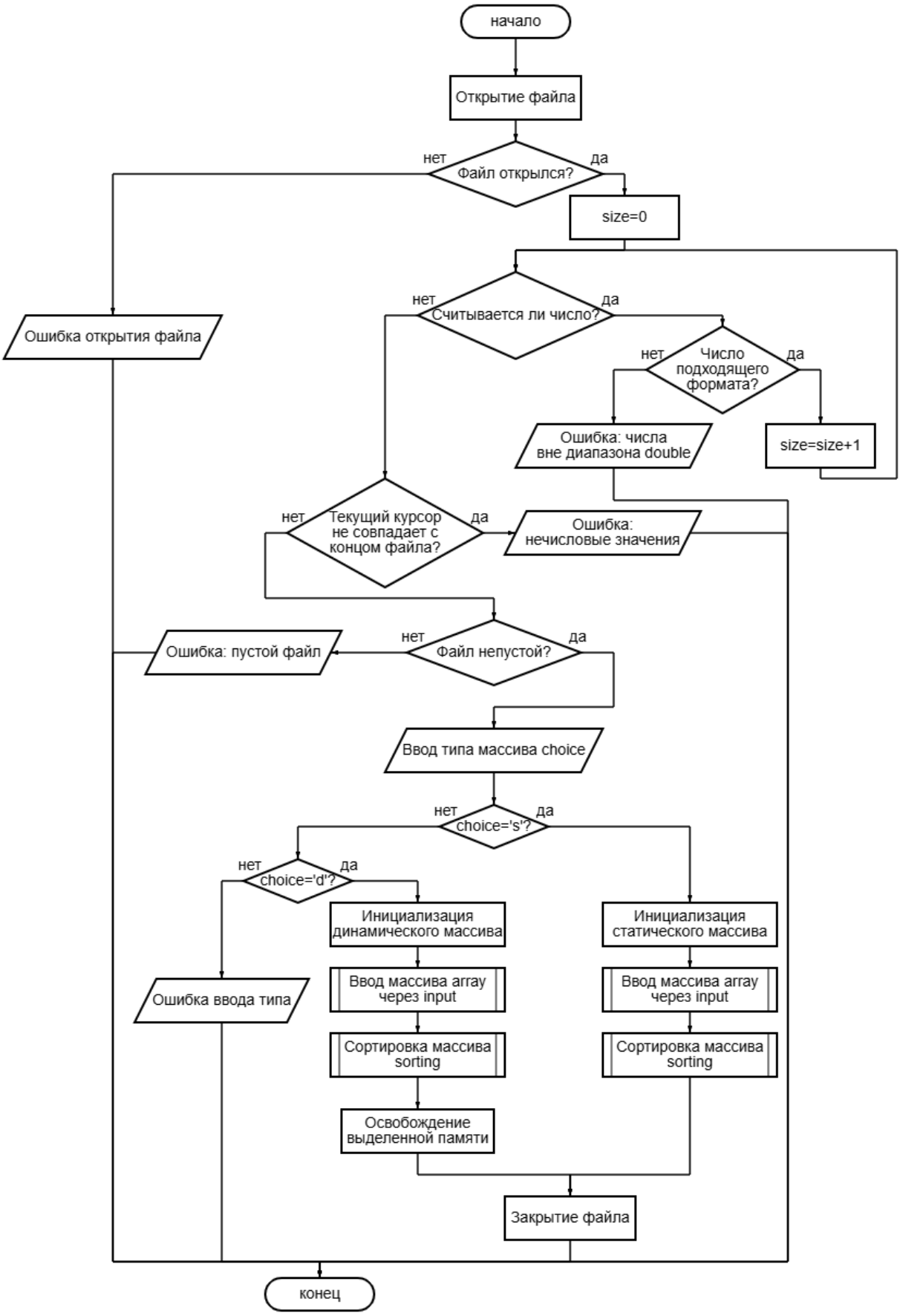

Рисунок 3 – Блок-схема основной части программы

КОД

Ниже приведен код, написанный для сортировки слиянием чисел, подаваемых через файл в статический или динамический массив по выбору пользователя, по указанной выше блок-схеме. Программа реализована на языке C++.

```
#include <iostream>
#include <fstream>
#include <string>
#include <cmath>
#include <limits>
using namespace std;
void print(int n, double *nums) {// Вывод массива
	for (int i = 0; i < n; ++i) {
		cout << nums[i] << " ";
	}
	cout << endl;
}
void input(string file_path,int size, double *array){// Чтение чисел из файла
	double value;
	ifstream file(file_path);
	for (int i = 0; i < size; ++i) {
		file >> array[i];
	}
}
void merge(double *array, int left, int mid, int right){// Слияние двух частей
	int it1=0, it2=0;
	double* result = new double[right-left];  // Вспомогательный динамический массив
	// Добавление меньших чисел из двух частей
	while(left+it1<mid and mid+it2<right){
		// Меньшее число из левой части
		if(array[left+it1]<array[mid+it2]){
			result[it1+it2]=array[left+it1];
			it1+=1;
		}
```

```
            // Меньшее число из правой части
            else {
                result[it1+it2]=array[mid+it2];
                it2+=1;
            }
        }
        // Добавление оставшихся чисел из одной из частей
        while(left+it1<mid){
            result[it1+it2]=array[left+it1];
            it1+=1;
        }
        while(mid+it2<right){
            result[it1+it2]=array[mid+it2];
            it2+=1;
        }
        // Перезаписываем исходный массив
        for (int i=0;i<it1+it2;i++){
            array[left+i]=result[i];
        }
        // Освобождение памяти из-под вспомогательного массива
        delete[] result;
}
void sorting(int n, double *array){
        // Вывод содержимого неотсортированного массива
        cout << "Unsorted array:" << endl;
        print(n,array);
        // Сортировка
        // Определяем срезы массива и проходимся сортировкой по ним
        for(int i=1;i<n+1;i*=2){
            for(int j=0;j<n-1;j+=2*i){
                merge(array,j,j+i,min(j+2*i,n));
            }
        }
```

```
        // Вывод содержимого отсортированного массива
        cout << "Sorted array:" << endl;
        print(n,array);
}
int main()
{
        string file_path="data.txt";
        // Открываем файл для чтения
        ifstream file(file_path);
        // Проверка на ошибку при открытии
        if (!file.is_open()) {
                cerr << "Error opening file" << endl;
                return 1;
        }
        // Определяем количество чисел в файле
        long double tempnum;
        int size=0; // Инициализация размерности массива
        while (file >> tempnum) {
                // Проверка на корректность данных
                if(tempnum<numeric_limits<double>::max() and
        tempnum>numeric_limits<double>::lowest()){
                        size++; // Увеличиваем количество чисел
                        continue;
                }
                cout<<"Error: numbers out of the range of type double";
                return 1;}
        // Проверка на ввод всех символов
        file.clear(); // Сброс возможных ошибок
        int cur=file.tellg(); // Сохраняем позицию курсора
        file.seekg(0, ios::end); // Перемещаем курсор в конец файла
        if(cur!=file.tellg()) { //Сравниваем сохраненный курсор с концом файла
                cout<<"Error: non-numeric data";
                return 1;}
```

```
        // Проверка на наличие данных
         if(size==0){
                cout<<"Error: empty file";
                return 1;}
        // Ввод типа массива
        string choice;
        cout << "Select array type (s - static, d - dynamic): ";
        cin >> choice;
        // Обработка выбора пользователя
        if(choice=="s") {// Статический массив
                double array[size]; // Ограничение на размер массива
                input(file_path,size,array);
                sorting(size,array);
        }
        else if(choice=="d") {// Динамический массив
                double* array = new double[size]; // Выделение памяти
                input(file_path,size,array);
                sorting(size,array);
                delete[] array;// Освобождение выделенной памяти
        }
        else{ //Ошибка выбора типа массива
                cerr << "Error: incorrect input";
                return 1;
        }
        //Закрываем файл
        file.close();
        return 0;
}
```

ТЕСТИРОВАНИЕ

Проведено тестирование программы с помощью различных входных данных. Был использован компилятор C++ 3.4.2 и среда разработки Embarcadero Dev-C++ 6.3

1. Статический массив с разнообразными числами (1e+2 -200 300 -9 4)

```
Select array type (s - static, d - dynamic): s
Unsorted array:
100 -200 300 -9 4
Sorted array:
-200 -9 4 100 300
```

Рисунок 4 – Тест №1

2. Динамический массив с разнообразными числами (1e+2 -200 300 -9 4)

```
Select array type (s - static, d - dynamic): d
Unsorted array:
100 -200 300 -9 4
Sorted array:
-200 -9 4 100 300
```

Рисунок 5 – Тест №2

3. Массив, где значения по возрастанию (-9 -5 0 6 34)

```
Select array type (s - static, d - dynamic): s
Unsorted array:
-9 -5 0 6 34
Sorted array:
-9 -5 0 6 34
```

Рисунок 6 – Тест №3

4. Массив, где значения по убыванию (200 100 9 -4 -5 -67)

```
Select array type (s - static, d - dynamic): d
Unsorted array:
200 100 9 -4 -5 -67
Sorted array:
-67 -5 -4 9 100 200
```

Рисунок 7 – Тест №4

5. Динамический массив (20 -4 52 85 6 49 1 4 28 -2 03 -2 7753 -3456 -1647 782)

```
Select array type (s - static, d - dynamic): d
Unsorted array:
20 -4 52 85 6 49 1 4 28 -2 3 -2 7753 -3456 -1647 782
Sorted array:
-3456 -1647 -4 -2 -2 1 3 4 6 20 28 49 52 85 782 7753
```

Рисунок 8 – Тест №5

6. Статический массив (20 -4 52 85 6 49 1 4 28 -2 03 -2 7753 -3456 -1647 782)

```
Select array type (s - static, d - dynamic): s
Unsorted array:
20 -4 52 85 6 49 1 4 28 -2 3 -2 7753 -3456 -1647 782
Sorted array:
-3456 -1647 -4 -2 -2 1 3 4 6 20 28 49 52 85 782 7753
```

Рисунок 9 – Тест №6

7. Файл с нечисловыми символами (1e+2 -20sdf0 300 -9 4)

```
Error: non-numeric data
```

Рисунок 10 – Тест №7

8. Файл с числами, превышающими диапазон типа double (1e+3000 -200 300 -9 4)

```
Error: incorrect data
```

Рисунок 11 – Тест №8

9. Пустой файл

```
Error: empty file
```

Рисунок 12 – Тест №9

10. Неправильный ввод типа массива

```
Select array type (s - static, d - dynamic): ss
Error: incorrect input
```

Рисунок 13 – Тест №10

ЗАКЛЮЧЕНИЕ

Была изучена сортировка слиянием и написана блок-схема алгоритма программы, которая запрашивает у пользователя тип массива (динамический или статический) и сортирует числа из файла. На языке C++ написан код программы с выводом неотсортированного и отсортированного массива, обработкой ошибок и интерфейсом для пользователя. Программа успешно протестирована на различных наборах данных.

Все задачи выполнены и цель достигнута.