Лабораторная работа 3
Сортировка

Цель работы

Изучить особенности работы со структурами, рекурсией и шаблонами в C++.

Стандарт языка

C++17 и новее.

Описание

Программа должна сортировать массивы различных типов данных, используя алгоритм быстрой сортировки. Должна поддерживаться возможность сортировки по возрастанию и убыванию.

Типы данных

Программа должна сортировать данные 3 типов:

1. int
2. float
3. phonebook

Пример записи структуры phonebook в файле:

<фамилия><*пробел*><имя><*пробел*><отчество><*пробел*><телефон>

где:

- <фамилия>, <имя>, <отчество> – строки не длиннее 20 символов, не содержат пробелов, табуляции, переводов строк или ноль-символов;
- <телефон> – целое неотрицательное число, меньшее 10^{11}.

Сортировка структур проводится в следующем порядке (если совпадают первые элементы, то сравниваются следующие):

фамилия->имя->отчество->телефон.

Каждое поле phonebook должно храниться отдельным полем структуры/класса.
Числа сортируются в арифметическом порядке, однобайтовые строки символов сортируются по коду символа (регистрозависимо).

Реализация по файлам

Весь код программы должен быть реализован в нескольких файлах.

Файл	**Что должно быть в файле (порядок условный)**
main.cpp	Подключение заголовочных файлов стандартной библиотеки, необходимых для этого файла Подключение своих заголовочных файлов [Optional] Свои функции, которые относятся к `main` `int main(`*аргументы*`)`

quicksort.h	Подключение заголовочных файлов стандартной библиотеки, необходимых для этого файла
	
	Подключение своих заголовочных файлов
	
	Определение шаблона функции сортировки со следующими аргументами шаблона
	`template <typename T, bool descending>`
	`void quicksort(`*ваши_аргументы*`)`

phonebook.h	[Optional] Подключение заголовочных файлов стандартной библиотеки, необходимых для этого файла [Optional] Подключение своих заголовочных файлов Объявление структуры/класса `phonebook`

	Сигнатуры функций/операторов для работы с phonebook
phonebook.cpp	[Optional] Подключение заголовочных файлов стандартной библиотеки, необходимых для этого файла Подключение своих заголовочных файлов Реализация функций/методов/операторов для работы с phonebook

Формат аргументов командной строки

Аргументы программе передаются через командную строку:

`cpp1 <имя_входного_файла> <имя_выходного_файла>`

Формат входного файла

Во входном файле в первой строке указан **тип данных**. Допустимые значения (без кавычек): “int”, “float”, “phonebook”.

Во второй строке указан **режим сортировки**. Допустимые значения: “ascending”, “descending”.

На третьей строке указано кол-во значений в файле.

После этого на каждой строке задан элемент указанного типа. По одному значению на строку, каждая строка завершается символом перевода строки.

Пример входного файла:

phonebook ascending 2 aa bb cc 255 Aa bB cc 265	int descending 2 5 3

Требования к программе

Программа должна:

1. быть написана на C++ по заданному стандарту;
2. выполнять поставленную в ТЗ задачу;
3. не использовать внешние библиотеки;
4. всегда корректно освобождать память;
5. всегда корректно закрывать файлы;
6. обрабатывать ошибки:
    a. файл не открылся;
    b. не удалось выделить память;
    c. на вход передано неверное число аргументов командной строки.

    В этих случаях необходимо выдавать сообщение об ошибке и корректно завершаться с ненулевым кодом возврата (см. `"return_codes.h"`);
7. никогда ничего не писать **в поток вывода**;
8. выводить сообщения об ошибке в **поток вывода ошибок**.

Ограничения

1. Запрещено использование exit(...) в коде.
2. Запрещено создавать VLA-массивы (но можно VLA-указатели).
3. Ограничивается использование глобальных переменных (кроме констант) - необходимость их использования вы должны обосновать на защите. Ваш код должен быть максимально приспособлен к переносимости в другие проекты и/или использованию другими разработчиками.
4. Запрещается подключать системные библиотеки через `#include "..."`.

5. Запрещается использовать setlocale(...). Учимся писать небольшие комментарии пользователю по-английски.
6. Запрещается использовать system(“pause”).
7. Если вы создаёте свои макросы, то их название не должно быть “DEBUG”, “_DEBUG”, “NDEBUG” и прочие, определяемые компиляторами имена. Допустимы любые другие названия из заглавных букв и символов подчёркивания.

Комментарии по проверкам

Дабы избежать ситуации “Я же отправил что-то, почему мне не зафиксировал дедлайн?”, а на деле там открытие и закрытие файлов, то в этот раз есть ограничения, что должно быть отправлено на проверку (из минимума).

Отправка 1: Работающий на значениях типа int алгоритм (можно опустить шаблон или явно задать специализацию под int на этом этапе). Проверки на открытие входного файла, на выделение памяти под массив данных. Сортировка, вывод результата.

Отправка 2: Реализация алгоритма сортировки с шаблоном, должно работать на всех типах данных по заданию. Должны быть все проверки.

Отправка 3: Багфиксим реализации, добираем баллы.

Если во входном файле вы встречаете тип данных, с которым пока что не умеете работать, то нужно завершать программу с ошибкой NOT_IMPLEMENTED.

Полезные check-листы и ссылки:

- Проверки

- Отправка на GitHub
- Про критерии оценивания
- Суперпопытка