**Министерство образования и науки Российской Федерации**

**Пермское федеральное государственное бюджетное образовательное учреждение высшего образования «Пермский Национальный Исследовательский Политехнический Университет»**

**Электротехнический факультет**

**Кафедра «Информационные технологии и автоматизированные системы»**

**ОТЧЁТ**

По лабораторной работе №16 на тему

«Алгоритмы внешних сортировок»

Вариант №11

Выполнил студент группы РИС-20-1б

Шумилов Лев Сергеевич

Проверил доцент кафедры ИТАС

Полякова О.А.

Пермь 2021

**Цель работы** –научиться работать и использовать алгоритмы внешней сортировки таких как:

1. Сортировка естественным слиянием;
2. Сортировка двоичным слиянием;
3. Сортировка многофазным слиянием;

**Постановка задачи**

Задача – реализовать на языке C++ функции внешних сортировок и продемонстрировать их работу в программе (в главной функции main).

* Составить свою структуру данных как один элемент;
* Прочитать из файла данные и записать их в массив/список данных, которые необходимо отсортировать;
* Продемонстрировать работу функций внешних сортировок;

Структура «Герой»:

* Полное имя (ФИО);
* Класс героя;
* Год рождения;

**Анализ задачи**

Для решения задачи были использованы следующие средства:

1. Язык программирования C++ (Microsoft Visual C++)
2. Текстовый редактор Microsoft Visual Studio Code
3. Система контроля версий Git

**Реализация программы:**

Для использования операторов ввода-вывода необходимо подключить следующие заголовочные файлы:

1. iostream, содержащий объекта потоков ввода-вывода: std::cin и std::cout;
2. fstream, который содержит классы необходимые для чтения из файлов или записи данных в них;
3. string, позволяющий работать со строками;
4. vector, необходимый для работы с расширяемыми массивами – векторами;

Также, для удобства было подключено пространство имён std, который в себе содержит описанные выше потоки ввода-вывода. Заголовочные файлы подключаются при помощи директивы #include.

#include <iostream>

#include <vector>

#include <string>

using namespace std;

Сама структура, описанная в постановке задача объявляется следующим образом:

struct Hero

{

string FullName;

string HeroRole;

int BirthYear;

};

Поскольку в условии задачи требуется осуществлять ввод данных из файла, необходимо реализовать функцию, отвечающую за ввод данных с файла и дальнейшую запись в вектор.

Функция в качестве аргумента принимает строку, содержащую путь к файлу, с которого будет производиться чтение.

vector<Hero> ReadHeroFromFile(string filePath)

Далее идёт объявление вектора, который будет накапливаться и в конце будет возвращаться и класс, отвечающий за чтение из файла – ifstream. Затем идёт проверка на успешность обращения к файлу: если обращение произошло успешно, то продолжается выполнение, в противном случае выводится сообщение об ошибке и выполнение прекращается.

vector<Hero> heros;

ifstream reader(filePath);

if (reader.fail())

cout << "error" << endl;

else

{ /\* дальнейшие шаги \*/ }

Далее, пока не был достигнут конец файла, через функцию getline осуществляется построчное чтение из файла в каждый элемент структуры (данные в файле расположены в соответствии с элементами структуры).

После того, как один элемент – структура успешно записывается, данный элемент добавляется в конец вектора.

while (!reader.eof())

{

Hero oneHero;

string birthYear;

getline(reader, oneHero.FullName);

getline(reader, oneHero.HeroRole);

getline(reader, birthYear);

oneHero.BirthYear = stoi(birthYear);

heros.push\_back(oneHero);

}

После того как все элементы были записаны в вектор, его необходимо вернуть как значение функции.

return heros;

Также, необходимо осуществлять вывод всех элементов на экран. Для этого были реализованы две функции:

1. Основная функция, которые выводит значение всех полей элемента на экран;
2. Внешняя функция, которая через цикл for вызывает основную функцию вывода;

Основная функция в качестве аргумента принимается сам элемент, данные которого необходимо вывести на экран и его индекс – порядковый номер.

void PrintHero(Hero hero, int orderNumber)

{

cout << "Hero #" << orderNumber + 1 << ";\n";

cout << " - Full name: " << hero.FullName << ";\n";

cout << " - Hero role: " << hero.HeroRole << ";\n";

cout << " - Birth year: " << hero.BirthYear << ";\n\n";

}

Внешняя же функция в качестве аргумента принимает сам вектор, выводит на экран количество элементов и в цикле for вызывает основную функцию вывода. В основной функции main будет вызываться именно внешняя функция.

void PrintHeros(vector<Hero> heros)

{

cout << "Hero count = " << heros.size() << ";\n\n";

for (int i = 0; i < heros.size(); i++)

PrintHero(heros[i], i);

}

**Функция main:**

Основная функция выполняет следующие действия:

1. Вызывает функцию чтения данных с файла;
2. Выводит на экран полученный вектор;