**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МО ЭВМ**

**ОТЧЕТ**

**по лабораторной работе №4**

**по дисциплине «Алгоритмы и структуры данных»**

**Тема: Сортировки**

|  |  |  |
| --- | --- | --- |
| Студент гр. 9382 |  | Савельев И.C. |
| Преподаватель |  | Фирсов М.А. |

Санкт-Петербург

2020

**Цель работы.**

Реализовать сортировку выбором и сортировку выбором с поиском минимума и максимума на языке С++.

**Задание.**

Вариант 1

1. Сортировка выбором; сортировка выбором с одновременным выбором максимума и минимум.

**Описание функций.**

string readFile(const string& file\_name), получает на вход константную ссылку на строку с названием файла file\_name. Внутри функции создается экземпляр класса ifstream f, которому передается название файла и экземпляр класс stringstream ss. Затем с помощью метода класса rdbuf() в ss передается указатель на буфер filebuf\*. В конце функция с помощью метода str() возвращает строку считанную из файла.

void printArray(vector<T>& array), получает на вход вектор и с помощью функции copy() и итератора вывода ostream\_iterator<T>(cout, " ") выводит его содержимое.

void sortMass1(vector<T>& array), получает на вход вектор, с помощью внешнего цикла перебирает каждый элемент вектора кроме последнего на каждой итерации наименьшим считается первый элемент подмассива, затем с помощью внутреннего цикла ищется наименьший элемент подмассива, если такой находится, то первый элемент подмассива и наименьший в нем меняются местами с помощью функции swap(), затем выводятся промежуточные данные, частично отсортированный массив с помощью функции printArray() и наименьший элемент в подмассиве.

void sortMass2(vector<T>& array), получает на вход вектор, с помощью внешнего цикла перебирается каждый элемент вектора, перед выполнением внутреннего цикла наименьшим считается первый элемент подмассива, а наибольшим последний. Затем в первом внутреннем цикле ищется наименьший элемент в подмассиве и с помощью функции swap() меняется местами с первым элементом подмассива. Аналогично во втором внутреннем цикле ищется наибольший элемент подмассива и меняется местами с последним элементом подмассива с помощью функции swap(). После чего выводятся ромежуточный данные, частично отсортированный массив с помощью функции printArray() и наименьший и наибольший элемент в подмассиве.

**Выполнение работы.**

В начале выполнения программы с помощью функции string readFile(), содержимое файла считывается в экземпляр класса stringstream ss, затем с помощью библиотечной функции copy содержимое ss переноситься в вектор mass. После чего создаются два вектора mass1 и mass2 равные mass. Затем выводится содержимое введенного пользователем массива с помощью функции printArray() и выполняется сортировка массива mass1 c помощью sortMass1(). Аналогичные действия происходят с mass2.

**Алгоритм.**

Сортировка выбором

Ищется наименьший элемент в массиве и перемещается на первое место. Затем ищется второй наименьший элемент и перемещается уже на второе место после первого наименьшего элемента. Этот процесс продолжается до тех пор, пока в массиве не закончатся не отсортированные элементы. Число проходов внешним циклом по массиву равно N-1, так как последний элемент уже будет отсортирован к моменту завершения обхода. Преимущество данного алгоритма - это простота его реализации. Недостаток - это его эффективность O(n^2).

Сортировка выбором с поиском минимума и максимума

Ищется наименьший и наибольший элементы в массиве, затем наименьший перемещается на первое место, а наибольший на последнее. После чего ищется второй наименьший и наибольший элементы в массиве и перемещаются уже на второе место после первого наименьшего элемента и на предпоследнее место соответственно. Число проходов по массиву внешним циклом равно N/2. Преимущество данного алгоритма - это простота его реализации, а его преимущество над обычной сортировкой, то что не отсортированная часть при каждой итерации внешнего цикла уменьшается сразу на два элемента, но так как количество внутренних циклов, свапов и сравнений равно двум - это лишь незначительно увеличивает скорость. Его главный недостаток - это его эффективность O(n^2).

**Тестирование.**

Результаты тестирования представлены в табл. 1.

Таблица 1 - Результаты тестирования

|  |  |  |
| --- | --- | --- |
| № п/п | Входные данные | Выходные данные |
| 1 | 1 | 1 |
| 2 | 4 6 3 | 3 4 6 |
| 3 | 3 4 6 | 3 4 6 |
| 4 | 45 32 75 385 24 1 | 1 24 32 45 75 385 |
| 5 | 31 73 14 41 4 2 774 14 41 461 471 7416 | 2 4 14 14 31 41 41 73 461 471 774 7416 |
| 6 | a d a f a | incorrect work |
| 7 |  | incorrect work |
| 8 | !@#$% | incorrect work |

**Выводы.**

Была успешно реализована на языке С++ сортировка выбором и сортировка выбором с поиском минимума и максимума.

**Исходный код программы.**

#include <algorithm>

#include <iostream>

#include <iterator>

#include <sstream>

#include <fstream>

#include <string>

#include <vector>

#define FILE "/home/indiora/C++/Aisd/read.txt"

using namespace std;

string readFile(const string& file\_name) {

ifstream f(file\_name); // Экземпляр класса

stringstream ss;

ss << f.rdbuf(); // Возвращает указатель на потоковый буфер

return ss.str(); // Возвращает строку

}

template <typename T>

void printArray(vector<T>& array) {

// os... записывает элементы в поток вывода

copy(begin(array), end(array), ostream\_iterator<T>(cout, " "));

std::cout << '\n';

}

template <typename T>

void sortMass1(vector<T>& array) {

// Перебираем каждый элемент массива кроме последнего

for (int startIndex = 0; startIndex < array.size() - 1; ++startIndex) {

// Наименьшим считается 1 элеменент

int smallestIndex = startIndex;

// Ищем элемент поменьше в остальной части массива

for (int currentIndex = startIndex + 1; currentIndex < array.size(); ++currentIndex) {

if (array[currentIndex] < array[smallestIndex])

smallestIndex = currentIndex;

}

// Меняем местами первый элемент подмассива и наименьший в нем

swap(array[startIndex], array[smallestIndex]);

// Выводим массив

std::cout << "Sort number " << startIndex << " : ";

printArray(array);

// Выводим промежуточное значение

std::cout << '\t' << "Smallest number: " << array[startIndex] << '\n' << '\n';

}

}

template <typename T>

void sortMass2(vector<T>& array) {

// Перебираем каждый элемент массив

for(int i = 0; i < ((array.size())/2); i++) {

int smallestIndex = i;

int biggestIndex = array.size() - 1 - i;

// Ищем наименьший элеммент подмассива

for(int j = i + 1; j < array.size(); j++) {

if(array[j] < array[smallestIndex])

smallestIndex = j;

}

// Меняем местами первый элемент подмассива и наименьший в нем

swap(array[i], array[smallestIndex]);

// Ищем наибольший элеммент подмассива

for(int k = i + 1; k < array.size() - i; k++) {

if(array[k] > array[biggestIndex])

biggestIndex = k;

}

// Меняем местами последний элемент подмассива и наибольший в нем

swap(array[array.size() - 1 - i],array[biggestIndex]);

// Выводим массив

std::cout << "Sort number " << i << " : ";

printArray(array);

// Выводим промежуточные значения

std::cout << '\t' << "Smallest number: " << array[i] << '\n';

std::cout << '\t' << "Biggest number: " << array[biggestIndex] << '\n' << '\n';

}

}

int main() {

stringstream ss(readFile(FILE)); // Вставляет строку в поток

vector<int> mass;

// Переносим значение из строки в массив

copy(istream\_iterator<int>(ss), {}, back\_inserter(mass)); // back\_inserter - добавление в конец вектора

// is... считывает элементы с входного потока

vector<int> mass1 = mass;

vector<int> mass2 = mass;

std::cout << "Sort by selection" << '\n';

std::cout << "Initial array: ";

printArray(mass1);

std::cout << '\n';

sortMass1(mass1);

std::cout << '\n';

std::cout << "Sort by selection with min and max search" << '\n';

std::cout << "Initial array: ";

printArray(mass2);

std::cout << '\n';

sortMass2(mass2);

}