# Практическая работа № 8

**ФАЙЛЫ**

**Постановка задачи**

Создать программные модули с операциями над двоичными и текстовыми файлами для выполнения задания варианта. Двоичный файл состоит из записей определенной структуры. Записи имеют ключ, уникальный в пределах файла. Для выполнения варианта задания использовать потоковые файлы С++.

**1 - Преобразовать данные в двоичный файл**

**2 - Отобразить все записи двоичного файла**

**3 - Удаление записей по начальному фрагменту ключа**

**4 - Заменить не ключевой параметр у нескольких записей по ключу**

Перечисленные действия оформить в виде самостоятельных режимов работы созданного дека. Выбор режимов производить с помощью пользовательского меню.

Провести полное тестирование (всех режимов работы) программы на стеке, сформированном вводом с клавиатуры. Тест-примеры определить самостоятельно. Результаты тестирования в виде скриншотов экранов включить в отчет по выполненной работе.

Оформить отчет с подробным описанием созданного алгоритма, принципов программной реализации алгоритмов работы с файлами, описанием текста исходного кода и проведенного тестирования программы.

Сделать выводы о проделанной работе, основанные на полученных результатах.

1. **Описание алгоритма**

Алгоритм программы состоит из функции main и вызываемых в ней вспомогательных функций:

* **void read\_file** – функция считывания товаров файла.
* **void add\_in\_file** – функция добавления товара в файл.
* **int search\_in\_list\_id** – функция поиска товара в списке.
* **int search\_in\_list\_part\_id** – функция поиска по началу ключа.
* **void del\_elems\_part\_id** – функция удаления товаров по началу ключа.
* **void change\_elem\_id** – функция изменения товара по ключу.
* **void change\_elems –** функция именения нескольких товаров.
* **void file\_update() –** функция обновления файла.
* **void out\_list() –** функция вывода всех товаров из файла.

Для работы с файлами в языке c++ используется класс fstream.

ifstream для считывания файла, ofstream для его записи.

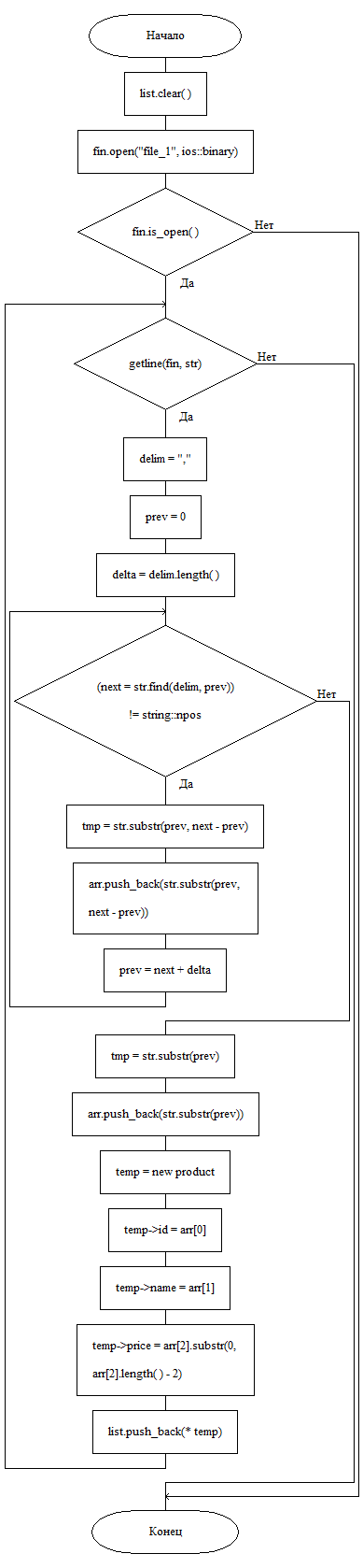


Рис.1 Схема алгоритма функции read\_file

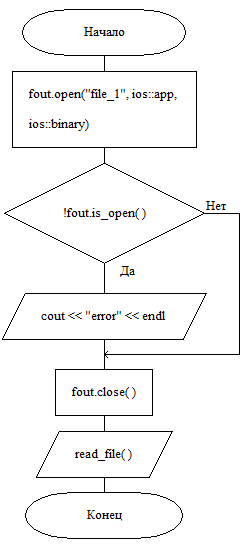


Рис.2 Схема алгоритма функции add\_in\_file

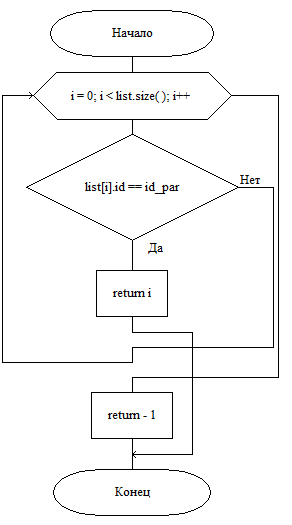


Рис.3 Схема алгоритма функции search\_in\_list\_id

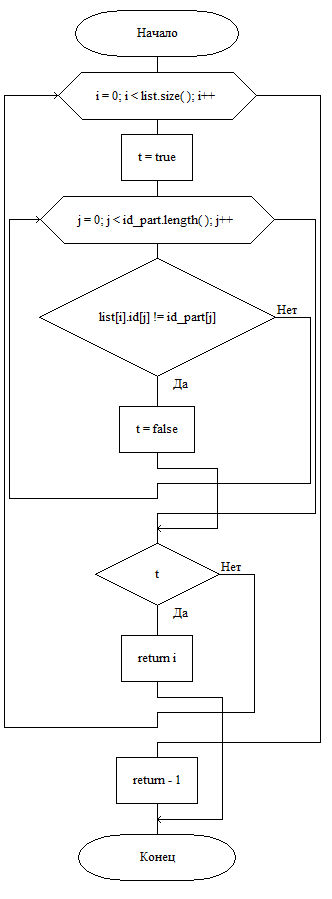


Рис.4 Схема алгоритма функции search\_in\_list\_part\_id

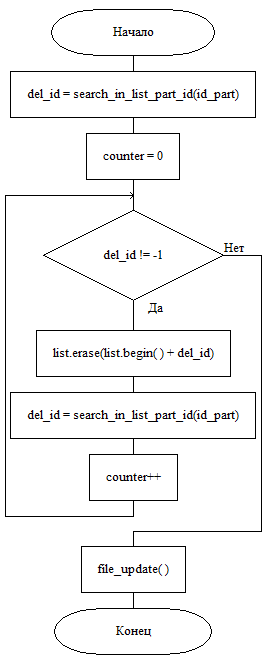


Рис.5 Схема алгоритма функции del\_elems\_part\_id

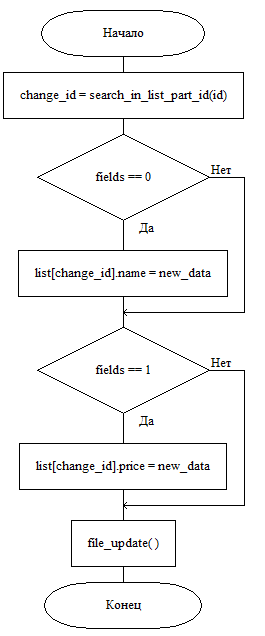


Рис.6 Схема алгоритма функции change\_elem\_id

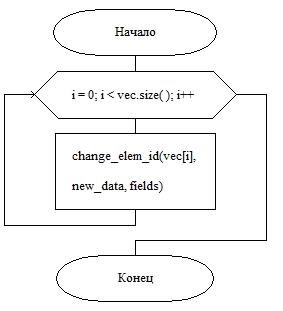


Рис.7 Схема алгоритма функции change\_elems

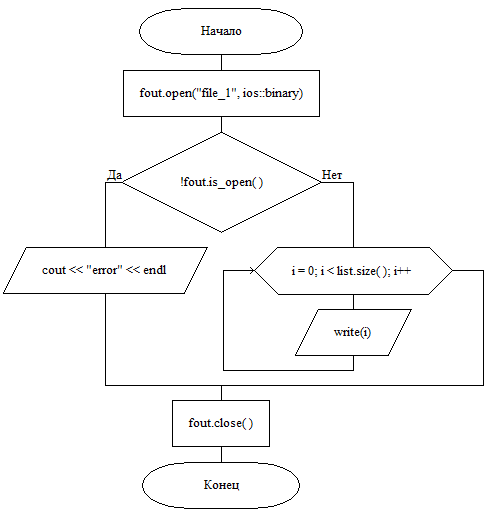


Рис.8 Схема алгоритма функции file\_update

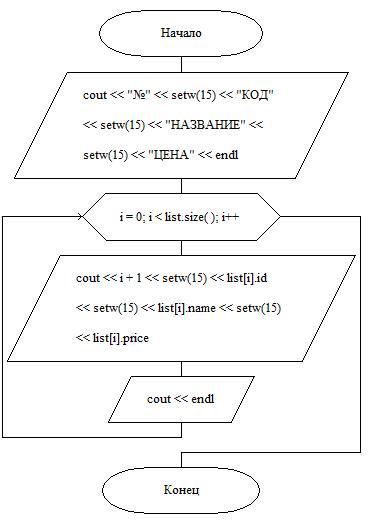


Рис.9 Схема алгоритма функции out\_list()

**Реализация алгоритма**

**Текст исходного кода программы**

**main.cpp**

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <iomanip>

using namespace std;

void file\_update();

struct product{

string id;

string name;

string price;

};

vector<product> list;

void read\_file() {

list.clear();

string str;

ifstream fin;

fin.open("file\_1", ios::binary);

if (fin.is\_open()) {

while (getline(fin, str))

{

vector<string> arr;

size\_t next;

string delim = ",";

size\_t prev = 0;

size\_t delta = delim.length();

while ((next = str.find(delim, prev)) != string::npos) {

string tmp = str.substr(prev, next - prev);

arr.push\_back(str.substr(prev, next - prev));

prev = next + delta;

}

string tmp = str.substr(prev);

arr.push\_back(str.substr(prev));

product\* temp = new product;

temp->id = arr[0];

temp->name = arr[1];

temp->price = arr[2].substr(0, arr[2].length()-1);

list.push\_back(\*temp);

}

}

}

void add\_in\_file(string s) {

ofstream fout;

fout.open("file\_1", ios::app, ios::binary);

if (!fout.is\_open()) cout << "error" << endl;

else {

fout << s << endl;

}

fout.close();

read\_file();

}

int search\_in\_list\_id(string id\_par) {

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

if (list[i].id == id\_par) {

return i;

}

}

return -1;

}

int search\_in\_list\_part\_id(string id\_part) {

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

bool t = true;

for (int j = 0; j < id\_part.length(); j++) {

if (list[i].id[j] != id\_part[j]) {

t = false;

break;

}

}

if (t) return i;

}

return -1;

}

void del\_elems\_part\_id(string id\_part) {

int del\_id = search\_in\_list\_part\_id(id\_part);

int counter = 0;

while (del\_id != -1) {

list.erase(list.begin() + del\_id);

del\_id = search\_in\_list\_part\_id(id\_part);

counter++;

}

file\_update();

cout << "Удалено товаров: " << counter << endl;

}

void change\_elem\_id(string id, string new\_data, int fields) {

int change\_id = search\_in\_list\_part\_id(id);

if(fields == 0){

list[change\_id].name = new\_data;

}

if (fields == 1) {

list[change\_id].price = new\_data;

}

file\_update();

}

void change\_elems(vector<string> vec, string new\_data, int fields) {

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

change\_elem\_id(vec[i], new\_data, fields);

}

cout << "Элементов изменено: " << vec.size() << endl;

}

void file\_update() {

ofstream fout;

fout.open("file\_1", ios::binary);

if (!fout.is\_open()) cout << "error" << endl;

else {

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

fout << list[i].id+","+ list[i].name + "," + list[i].price<<endl;

}

}

fout.close();

}

void out\_list() {

cout << "№" << setw(15) << "КОД" << setw(15) << "НАЗВАНИЕ" << setw(15) << "ЦЕНА" << endl;

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

cout <<i+1<<setw(15)<< list[i].id << setw(15) << list[i].name << setw(15) << list[i].price;

cout << endl;

}

}

int main()

{

setlocale(LC\_ALL, "Russian");

ofstream fout;

fout.open("file\_1", ios::binary);

if (!fout.is\_open()) cout << "error" << endl;

else {

fout << "";

}

fout.close();

while (true) {

cout << "Выберите команду:" << endl;

cout << "[1] - Добавить товар." << endl;

cout << "[2] - Удалить товары по началу ключа." << endl;

cout << "[3] - Изменить несколько товаров." << endl;

cout << "[4] - Вывести список товаров." << endl;

cout << "[5] - Завершить программу." << endl;

cout << "---->";

int ch = 0;

cin >> ch;

if (ch == 1) {

cout << "\_\_\_\_Введите НОМЕР, НАЗВАНИЕ и ЦЕНУ товара через запятые без пробелов" << endl;

cout << "-------->";

string new\_str;

cin >> new\_str;

add\_in\_file(new\_str);

continue;

}

if (ch == 2) {

cout << "\_\_\_\_Введите начало ключа" << endl;

cout << "-------->";

string new\_key;

cin >> new\_key;

del\_elems\_part\_id(new\_key);

continue;

}

if (ch == 3) {

cout << "\_\_\_\_Введите через пробел номера товаров (0 для завершения)" << endl;

cout << "-------->";

string new\_id = "";

vector<string> ids;

cin >> new\_id;

while (new\_id != "0") {

ids.push\_back(new\_id);

cin >> new\_id;

}

cout << "\_\_\_\_Введите новые данные и название поля для вставки" << endl;

cout << "-------->";

string data;

int field;

cin >> data >> field;

change\_elems(ids, data, field);

continue;

}

if (ch == 4) {

out\_list();

continue;

}

if (ch == 5) {

break;

}

}

}

1. **Тестирование программы**

Ниже представлен результат работы программы с бинарным файлом

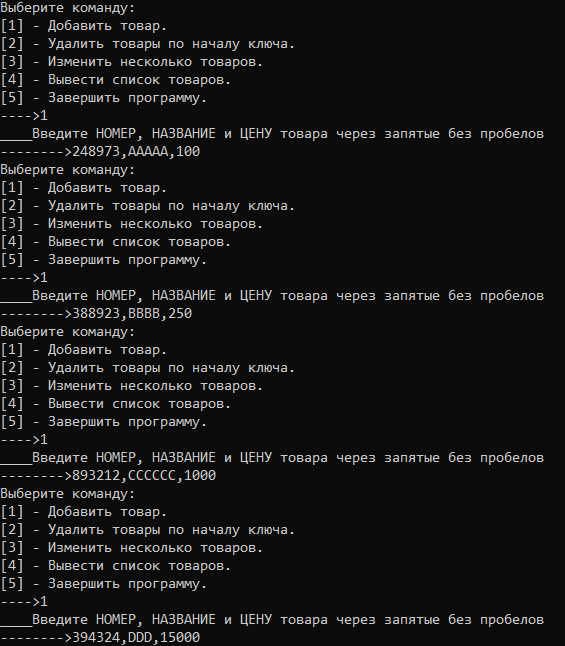


Рис.8 Скриншот добавления товаров в файл

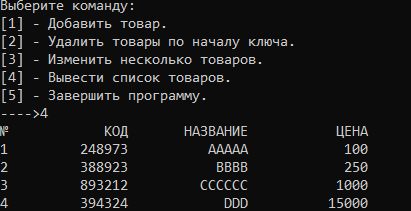


Рис.9 Скриншот вывода списка на экран

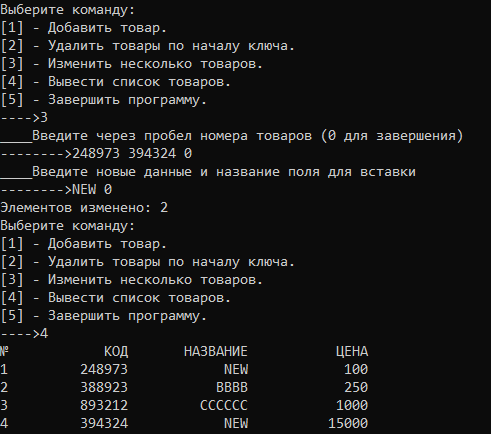


Рис.10 Скриншот изменения названия товаров по ключу и вывод

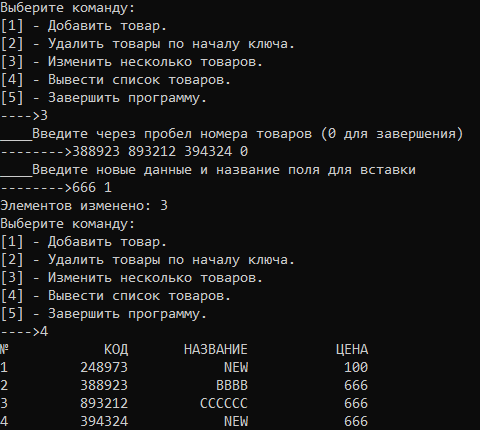


Рис.10 Скриншот изменения цены товаров по ключу и вывод

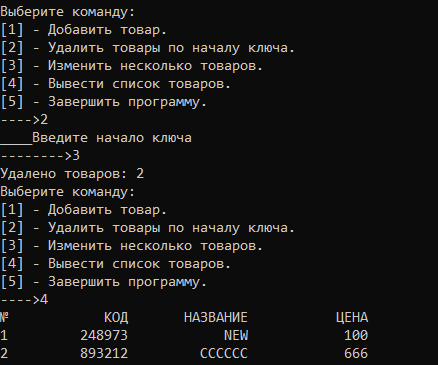


Рис.10 Скриншот удаления товаров по началу ключа

**Выводы**

1. В ходе работы была создана программа для работы с файлами.
2. Также были реализованы функции записи, чтения, удаления и изменения данных в файле.
3. Были изучены преимущества и недостатки хранения данных в файле:
4. Преимущества: данные в файлах могут храниться даже если устройство отключено от сети. В файлы можно записать гораздо больший объем данных чем в оперативную память
5. Недостатки: для работы необходимо постоянно обращаться к файловой системе.
6. Таким образом, была изучена работа алгоритмов обработки файлов

**Список используемых информационных источников**

1. Сыромятников В.П. Структуры и алгоритмы обработки данных, лекции, РТУ МИРЭА, Москва, 2020/2021 уч./год.
2. Документация по языку программирования С++, интернет-ресурс: <https://en.cppreference.com/w/> (Дата обращения – 02.11.2020)
3. Интегрированная среда разработки для языков программирования C и C++, разработанная компанией JetBrains - CLion / Copyright © 2000-2020 JetBrains s.r.o., интернет-ресурс: <https://www.jetbrains.com/clion/learning-center/> (Дата обращения – 02.11.2020).
4. ГОСТ 19.701-90 ЕСПД. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения. Интернет-ресурс: <http://docs.cntd.ru/document/gost-19-701-90-espd> (Дата обращения – 02.11.2020).
5. Описание файлов. интернет-ресурс: https://ru.wikipedia.org/wiki/Файл (Дата обращения – 02.11.2020).