Министерство образование Республики Беларусь

Учреждение образования «Полоцкий государственный университет»

Информационно-технологический факультет

Кафедра технологий программирования

Лабораторная работа №5

По дисциплина «Основы алгоритмизации и программирования»

Выполнил: Крутилин С.Д.

Группа 19-ИТ-1.

Проверил: Адамовский Е.

Новополоцк

2019 г.

**Структуры и перечисления**

**Вариант 2**

**Цель работы**: изучить библиотечные функции для работы с файлами. Освоить основные методы обработки файлов.

**ЗАДАНИЕ 1.** Функция new\_DB(), в которой будет создаваться новый файл базы данных и открываться для записи. Функция должна возвращать объект потока ввода для нового файла.

**Описание:** просим пользователя ввести название новой БД; создает объект класса ofstream с ранее указанным названием; открывает файл (или создает, если он отсутствует) для записи в конец; возвращает объект потока ввода для этого файла.

ofstream new\_DB() {

char file[256];

cout << "Введите название файла(с указанием формата): ";

gets\_s(file);

ofstream ofile(file, ios\_base::app);

if (ofile.is\_open()) {

cout << "Файл создан." << endl;

}

system("pause");

system("cls");

return ofile;

}

**ЗАДАНИЕ 2.** Функция remove\_DB(), удаляющая файл по названию. В функции должна быть реализована обработка ошибок.

**Описание:** просим пользователя указать название файла; удаляет файл с помощью remove() и делает проверку на выполнение; выводит соответствующее сообщение в консоль.

void remove\_DB() {

char file[256];

cout << "Введите название файла(с указанием формата): ";

gets\_s(file);

if (remove(file) != 0) {

cout << "Ошибка удаления файла" << endl;

}

else {

cout << "Файл успешно удалён" << endl;

}

system("pause");

system("cls");

}

**ЗАДАНИЕ 3.** Функция clear(), которая будет очищать динамическую память, выделенную для структур базы данных.

**Описание:** в buff возвращаем указатель на освобожденный блок памяти; обнуляем размер нашей базы данных; присваиваем нашем указателю на указатель на структуру указатель на освобожденный блок памяти (повеситься можно, если вникать).

**Результат работы программы:**

void clear() {

Contact\* buff = (Contact\*)realloc(information, 0);

kol = 0;

information = buff;

cout << "Данные успешно очищены" << endl;

system("pause");

system("cls");

}

**ЗАДАНИЕ 4.** Функция save\_DB(), которая будет сохранять данные из динамического массива структур в указанный файл.

**Описание:** просит пользователя ввести название файла для дальнейшей выгрузки данных; создает объект класса ofstream; создает файл в случае его отсутствия и очищает в случае его наличия; выводит размер нашего динамического массива структур, а затем выводит в каждой новой строке значение для каждого отдельного поля структуры (в цикле перебирается каждая структура в массиве).

void save\_DB() {

char file[256];

cout << "Введите название файла(с указанием формата): ";

gets\_s(file);

ofstream fout(file, ofstream::trunc);

fout << kol << endl;

for (int i = 0; i < kol; i++) {

fout << information[i].FIO << endl;

fout << information[i].city << endl;

fout << information[i].street << endl;

fout << information[i].house << endl;

fout << information[i].flat << endl;

fout << information[i].phone << endl;

fout << endl;

}

fout.close();

cout << "Запись в файл прошла успешно!" << endl;

system("pause");

system("cls");

}

**ЗАДАНИЕ 5.** Функция load\_DB(), которая будет выгружать данные из указанного файла в динамический массив структур.

**Описание:** просит пользователя ввести название файла для загрузки данных; создает объект класса ifstream(для считывания данных); осуществляет проверку на открытие файла; если файл удалось открыть считываем размер массива, преобразуем в целочисленное значение и присваиваем нашему указателю; в цикле перебираем каждую отдельную структуру и считываем поля в наш динамический массив; закрываем файл; выводим сообщение о успешном считывании файла.

void load\_DB() {

char file[50];

cout << "Введите название файла(с указанием формата): ";

gets\_s(file);

ifstream fin(file);

if (!fin.is\_open()) {

cout << "Ошибка загрузки файла." << endl;

}

else {

char ch[51];

fin.getline(ch, 51);

kol = atoi(ch);

resize();

for (int i = 0; i < kol; i++) {

fin.getline(information[i].FIO, 40);

fin.getline(information[i].city, 20);

fin.getline(information[i].street, 20);

fin.getline(ch, 11);

information[i].house = atof(ch);

fin.getline(ch, 11);

information[i].flat = atof(ch);

fin.getline(information[i].phone, 20);

}

fin.close();

cout << "Файл считан\n";

}

system("pause");

system("cls");

}

**Программа целиком:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <fstream>

#include <string>

#include <cstring>

#include <cstdlib>

#include <cstdio>

#include <conio.h>

#include <iomanip>

#include <ctime>

#include <Windows.h>

using namespace std;

struct Contact {

char FIO[40];

char city[20];

char street[20];

int house;

int flat;

char phone[13];

};

Contact\* information = NULL;

int num;

int kol = 0;

char Str[1000000000];

ofstream new\_DB();

void load\_DB();

void save\_DB();

void clear();

void remove\_DB();

void input();

void Tablitsa();

void list();

void resize();

int check\_input(int field);

void check\_kol();

void show\_menu() {

system("cls");

cout << " MENU \n" << endl;

cout << "1 - Ввод данных" << endl;

cout << "2 - Просмотр таблицы" << endl;

cout << "3 - Список" << endl;

cout << "4 - Новая БД" << endl;

cout << "5 - Сохранить БД" << endl;

cout << "6 - Загрузить БД" << endl;

cout << "7 - Очистить память" << endl;

cout << "8 - Удалить БД" << endl;

cout << "0 - Выход" << endl;

}

void menu() {

char ch;

show\_menu();

while (1) {

ch = \_getch();

if (ch == '1') {

system("cls");

input();

system("pause");

menu();

}

if (ch == '2') {

system("cls");

check\_kol();

Tablitsa();

system("pause");

menu();

}

if (ch == '3') {

system("cls");

check\_kol();

list();

system("pause");

menu();

}

if (ch == '4') {

system("cls");

new\_DB();

system("pause");

menu();

}

if (ch == '5') {

system("cls");

save\_DB();

system("pause");

menu();

}

if (ch == '6') {

system("cls");

load\_DB();

system("pause");

menu();

}

if (ch == '7') {

system("cls");

clear();

system("pause");

menu();

}

if (ch == '8') {

system("cls");

remove\_DB();

system("pause");

menu();

}

if (ch == '0') {

exit(0);

}

}

}

int main() {

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

system("mode con cols=138 lines=70");

menu();

return 0;

}

void input() {

int check = 0;

Contact person;

while (check != 1) {

system("cls");

cout << "Введите ФИО: ";

gets\_s(Str);

check = check\_input(8);

if (check != -1) {

check = check\_input(1);

}

}

strcpy\_s(person.FIO, Str);

check = 0;

while (check != 1) {

system("cls");

cout << "Введите город проживания: ";

gets\_s(Str);

check = check\_input(8);

if (check != -1) {

check = check\_input(2);

}

}

strcpy\_s(person.city, Str);

check = 0;

while (check != 1) {

system("cls");

cout << "Введите улицу проживания: ";

gets\_s(Str);

check = check\_input(8);

if (check != -1) {

check = check\_input(3);

}

}

strcpy\_s(person.street, Str);

check = 0;

while (check != 1) {

system("cls");

cout << "Введите номер дома проживания: ";

gets\_s(Str);

check = check\_input(4);

}

person.house = num;

check = 0;

while (check != 1) {

system("cls");

cout << "Введите номер квартиры проживания: ";

gets\_s(Str);

check = check\_input(5);

}

person.flat = num;

check = 0;

while (check != 1) {

system("cls");

cout << "Введите номер телефона: ";

gets\_s(Str);

check = check\_input(6);

if (check != -1) {

check = check\_input(0);

}

}

strcpy\_s(person.phone, Str);

check = 0;

kol++;

resize();

information[kol - 1] = person;

}

void list() {

if (kol == 0) {

cout << "Данные отсутствуют.";

}

else {

for (int i = 0; i < kol; i++) {

cout << "ФИО: " << information[i].FIO << endl;

cout << "Город: " << information[i].city << endl;

cout << "Улица: " << information[i].street << endl;

cout << "Дом: " << information[i].house << endl;

cout << "Квартира: " << information[i].flat << endl;

cout << "Номер: " << information[i].phone << endl;

cout << endl;

}

}

cout << endl;

system("pause");

system("cls");

}

void Tablitsa()

{

printf(" -----------------------------------------------------------------------------------------------------------------------------------------\n");

printf(" | № | Фамииля Имя Отчество | Город | Улица | Номер дома | Номер квартиры | Номер телефона |\n");

printf(" -----------------------------------------------------------------------------------------------------------------------------------------\n");

for (int i = 0; i < kol; i++)

{

printf(" |%5d|%40s|%20s|%20s|%12d|%16d|%16s|\n", i + 1,

information[i].FIO, information[i].city, information[i].street, information[i].house, information[i].flat, information[i].phone);

printf(" -----------------------------------------------------------------------------------------------------------------------------------------\n");

}

}

void resize() {

Contact\* buff = (Contact\*)realloc(information, sizeof(Contact) \* kol);

if (buff == NULL) {

cout << "Не удалось выделить память!" << endl;

exit(-1);

}

information = buff;

}

int check\_input(int field) {

int n = strlen(Str);

int pr = 0;

for (int i = 0; i < n; i++) {

if (Str[i] == ' ') pr++;

}

if (pr == n) {

return -1;

}

if (field == 0) {

if (n != 12) {

return -1;

}

else return 1;

}

if (field == 1) {

if (Str[0] == 0 || n > 40) {

return -1;

}

else return 1;

}

if (field == 2) {

if (Str[0] == 0 || n > 20) {

return -1;

}

else return 1;

}

if (field == 3) {

if (Str[0] == 0 || n > 20) {

return -1;

}

else return 1;

}

if (field == 4) {

if (Str[0] == 0 || n > 3 || atoi(Str) <= 0) {

return -1;

}

if (sscanf\_s(Str, "%d", &num) != 1) {

return -1;

}

else return 1;

}

if (field == 5) {

if (Str[0] == 0 || n > 3 || atoi(Str) <= 0) {

return -1;

}

if (sscanf\_s(Str, "%d", &num) != 1) {

return -1;

}

else return 1;

}

if (field == 6) {

int i = 0;

for (int i = 0; i < strlen(Str); i++)

{

if (Str[i] < '0' || Str[i] > '9')

{

return -1;

}

}

return 1;

}

if (field == 7) {

if (Str[0] == 0) {

return -1;

}

if (sscanf\_s(Str, "%d", &num) != 1) {

return -1;

}

if (num <= 0 || num > kol) {

return -1;

}

else return 1;

}

if (field == 8) {

int i = 0;

for (int i = 0; i < strlen(Str); i++)

{

if ((Str[i] < 'A' || Str[i] > 'Z') && (Str[i] < 'a' || Str[i] > 'z') && (Str[i] < 'А' || Str[i] > 'я') && Str[i] != ' ')

{

return -1;

}

}

return 1;

}

}

void check\_kol() {

if (kol == 0) {

cout << "Список пуст!" << endl;

system("pause");

menu();

}

}

////////////////

ofstream new\_DB() {

char file[256];

cout << "Введите название файла(с указанием формата): ";

gets\_s(file);

ofstream ofile(file, ios\_base::app);

if (ofile.is\_open()) {

cout << "Файл создан." << endl;

}

system("pause");

system("cls");

return ofile;

}

void remove\_DB() {

char file[256];

cout << "Введите название файла(с указанием формата): ";

gets\_s(file);

if (remove(file) != 0) {

cout << "Ошибка удаления файла" << endl;

}

else {

cout << "Файл успешно удалён" << endl;

}

system("pause");

system("cls");

}

void clear() {

Contact\* buff = (Contact\*)realloc(information, 0);

kol = 0;

information = buff;

cout << "Данные успешно очищены" << endl;

system("pause");

system("cls");

}

void save\_DB() {

char file[256];

cout << "Введите название файла(с указанием формата): ";

gets\_s(file);

ofstream fout(file, ofstream::trunc);

fout << kol << endl;

for (int i = 0; i < kol; i++) {

fout << information[i].FIO << endl;

fout << information[i].city << endl;

fout << information[i].street << endl;

fout << information[i].house << endl;

fout << information[i].flat << endl;

fout << information[i].phone << endl;

fout << endl;

}

fout.close();

cout << "Запись в файл прошла успешно!" << endl;

system("pause");

system("cls");

}

void load\_DB() {

char file[50];

cout << "Введите название файла(с указанием формата): ";

gets\_s(file);

ifstream fin(file);

if (!fin.is\_open()) {

cout << "Ошибка загрузки файла." << endl;

}

else {

char ch[51];

fin.getline(ch, 51);

kol = atoi(ch);

resize();

for (int i = 0; i < kol; i++) {

fin.getline(information[i].FIO, 40);

fin.getline(information[i].city, 20);

fin.getline(information[i].street, 20);

fin.getline(ch, 11);

information[i].house = atof(ch);

fin.getline(ch, 11);

information[i].flat = atof(ch);

fin.getline(information[i].phone, 20);

}

fin.close();

cout << "Файл считан\n";

}

system("pause");

system("cls");

}