Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа 5

По дисциплине «Основы алгоритмизации и программирования»

На тему «Объединения, перечисления, битовые поля»

Выполнил:

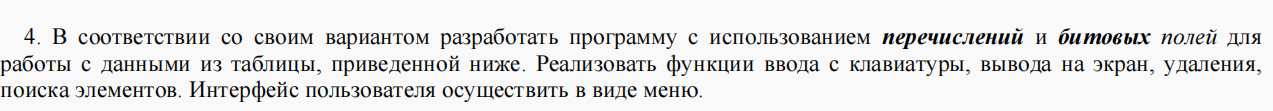
Студент 1 курса 6 группы

Романов Игорь Вячеславович

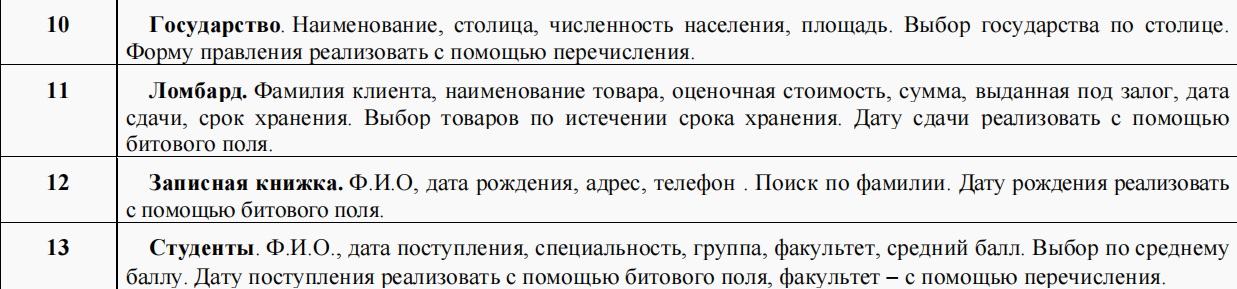
Преподаватель: асс. Андронова М.В.

2024, Минск

Задание 4



Вариант 12 (основной)



#include <iostream>

#include <Windows.h>

#include <cstring>

#define LIMIT 100 // ограничение по размеру массива структур

//прототипы функций

using namespace std;

void inputInfo();

void outputInfo();

void searchInfo();

void deleteInfo();

// структура

struct Notebook

{

char full\_name[80];

struct birth\_date {

unsigned day : 5;

unsigned month : 4;

unsigned year : 12;

} birthday;

char address[40];

char phone\_number[15];

};

//массив contact структуры Notebook

Notebook contact[LIMIT];

int current\_size = 0; // текущий размер массива

// главная фукнция

int main()

{

SetConsoleOutputCP(1251); // для распознавания кириллицы при записи в массив

SetConsoleCP(1251);

int choice;

do {

cout << "\nВведите вариант работы приложения: \n";

cout << "0. Выход из приложения\n";

cout << "1. Ввод элементов структуры с клавиатуры\n";

cout << "2. Вывод элементов структуры в консольное окно\n";

cout << "3. Удаление заданной структурированной переменной\n";

cout << "4. Поиск информации\n";

cin >> choice;

switch (choice)

{

case 1: { //Ввод элементов структуры с клавиатуры

inputInfo();

break;

}

case 2: { //Вывод элементов структуры в консольное окно

outputInfo();

break;

}

case 3: { //Удаление заданной структурированной переменной

deleteInfo();

break;

}

case 4: { //Поиск информации

searchInfo();

break;

}

case 0: { //Выход из программы

exit(0);

break;

}

default: {

cout << "Введено некорректное значение. " << endl;

break;

}

}

} while (choice != 0);

}

// ввод информации

void inputInfo()

{

if (current\_size < LIMIT) {

// фио

cout << "Введите ФИО: " << endl;

cin.ignore(); // очищаем буфер

cin.getline(contact[current\_size].full\_name, 50);

// др

int day, month, year;

cout << "Введите дату рождения в формате ДД ММ ГГГГ: " << endl;

cin >> day >> month >> year; //сначала записываем значения в переменные

contact[current\_size].birthday.day = day; // а потом добавляем их в битовые поля

contact[current\_size].birthday.month = month;

contact[current\_size].birthday.year = year;

cin.ignore(); // очищаем буфер после ввода даты рождения

// адрес

cout << "Введите адрес: " << endl;

cin.getline(contact[current\_size].address, 100);

// номер телефона

cout << "Введите номер телефона: " << endl;

cin >> contact[current\_size].phone\_number;

cin.ignore(); // очищаем буфер после ввода номера телефона

current\_size++;

cout << "Элемент структуры создан. " << endl;

}

else { //иначе выводим сообщение о переполнении

cout << "Структура переполнена!" << endl;

}

}

// вывод информации

void outputInfo()

{

for (int i = 0; i < current\_size; i++) { //поочередно выводим информацию о странах

cout << "\nЗапись №" << i + 1 << endl;

cout << "ФИО: " << contact[i].full\_name << endl;

cout << "Дата рождения: " << contact[i].birthday.day << "." << contact[i].birthday.month << "." << contact[i].birthday.year << endl;

cout << "Адрес: " << contact[i].address << endl;

cout << "Номер телефона: " << contact[i].phone\_number << endl;

}

}

// поиск информации

void searchInfo() {

char last\_name[20]; //переменная для поиска фамилии

cout << "Введите фамилию: " << endl;

cin >> last\_name;

bool flag = false; // по умолчанию считается, что фамилия не найдена

for (int i = 0; i < current\_size; i++) {

if (strstr(contact[i].full\_name, last\_name) != nullptr) { //если введенная фамилия найдена, то выводим информацию по ней

flag = true; // если фамилия найдена, то устанавливаем флажок

cout << "Информация по фамилии: " << last\_name << endl;

cout << "ФИО: " << contact[i].full\_name << endl;

cout << "Дата рождения: " << contact[i].birthday.day << "." << contact[i].birthday.month << "." << contact[i].birthday.year << endl;

cout << "Адрес: " << contact[i].address << endl;

cout << "Номер телефона: " << contact[i].phone\_number << endl;

}

}

if (!flag) { //иначе выводим сообщение об ошибке

cout << "Фамилия не была найдена" << endl;

}

}

// удаление информации по заданной переменной

void deleteInfo() {

char last\_name[20]; //переменная для поиска фамилии

cout << "Введите фамилию, которую хотите удалить из структуры: " << endl;

cin >> last\_name;

bool flag = false; // по умолчанию считается, что фамилия не найдена

for (int i = 0; i < current\_size; i++) {

if (strstr(contact[i].full\_name, last\_name) != nullptr) { //если введенная фамилия найдена, то удаляем ее из структуры

flag = true; // если фамилия найдена, то устанавливаем флажок

for (int j = i; j < current\_size - 1; j++) {

contact[j] = contact[j + 1];

}

current\_size--;

cout << "Структурная переменная очищена. " << endl;

}

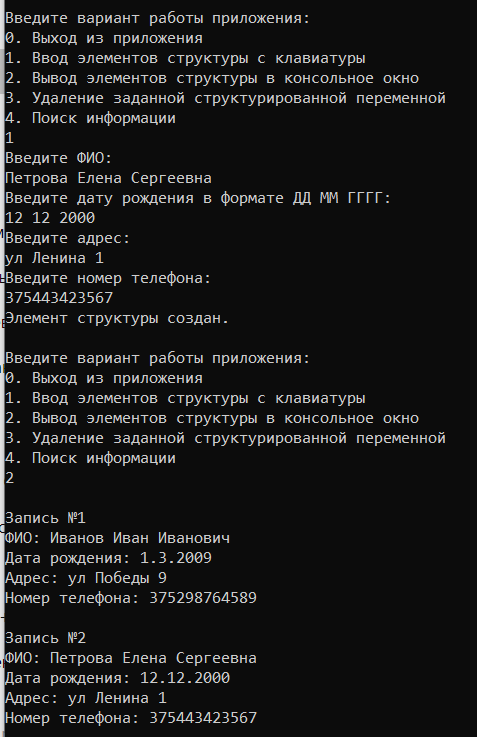
}

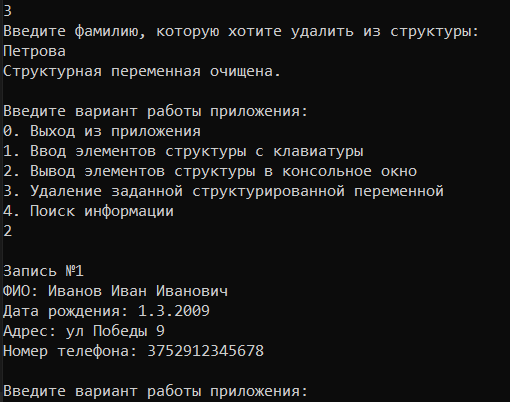
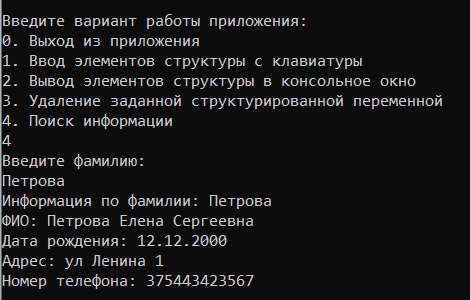
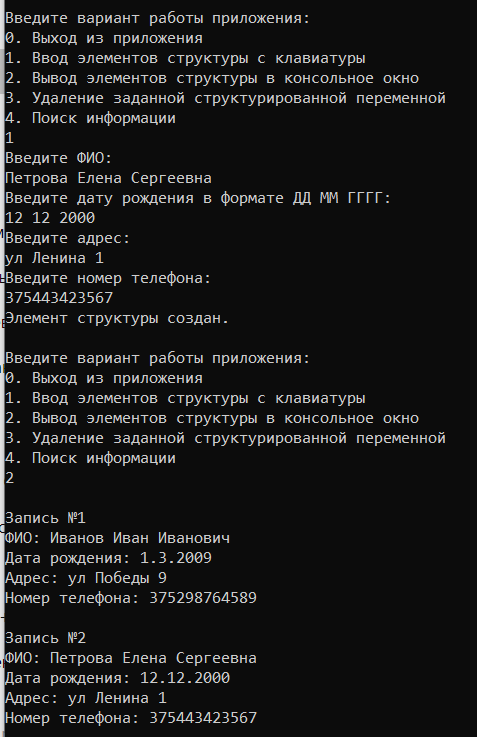
if (!flag) { //иначе выводим сообщение об ошибке

cout << "Фамилия не найдена";

}

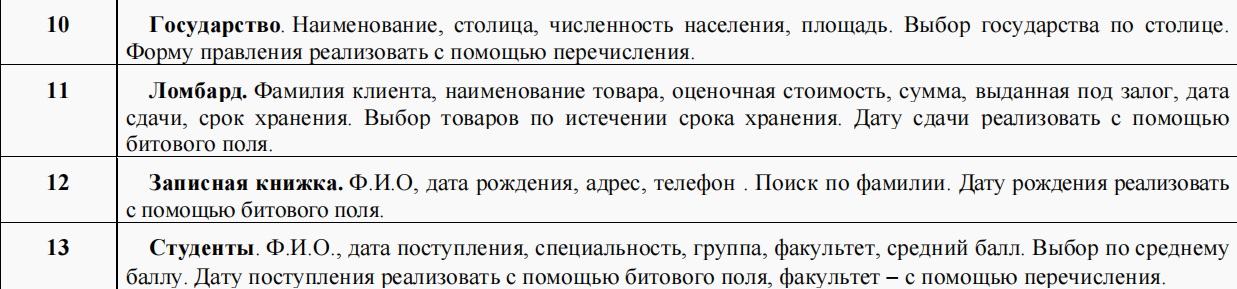
}





Доп.задания

Вариант 10



#include <iostream>

#include <Windows.h>

#include <cstring>

using namespace std;

#define LIMIT 100 // максимальное количество стран для массива структур

// перечисление

enum FormOfGovernment {

MONARCHY = 0,

REPUBLIC

};

// структура

struct Government {

char name[16];

char capital[16];

unsigned int population;

unsigned int area;

char president[16];

FormOfGovernment government;

};

Government country[LIMIT]; // массив country из объектов country

int current\_size = 0; // текущий размер массива

// прототипы функций

void inputInfo();

void outputInfo();

void searchInfo();

void deleteInfo();

int main() {

SetConsoleOutputCP(1251); // для распознавания кириллицы при записи в массив

SetConsoleCP(1251);

int choice;

do {

cout << "\nВведите вариант работы приложения: \n";

cout << "0. Выход из приложения\n";

cout << "1. Ввод элементов структуры с клавиатуры\n";

cout << "2. Вывод элементов структуры в консольное окно\n";

cout << "3. Удаление заданной структурированной переменной\n";

cout << "4. Поиск информации\n";

cin >> choice;

switch (choice) {

case 1: // Ввод элементов структуры с клавиатуры

inputInfo();

break;

case 2: // Вывод элементов структуры в консольное окно

outputInfo();

break;

case 3: // Удаление заданной структурированной переменной

deleteInfo();

break;

case 4: // Поиск информации

searchInfo();

break;

case 0: // Выход из программы

exit(0);

default:

cout << "Введено некорректное значение. " << endl;

break;

}

} while (choice != 0);

return 0;

}

void inputInfo() {

if (current\_size < LIMIT) { // если массив не выходит за пределы заданного, то вводим соотв. информацию

cout << "Введите название: ";

cin >> country[current\_size].name;

cout << "Введите столицу: ";

cin >> country[current\_size].capital;

cout << "Введите численность населения: ";

cin >> country[current\_size].population;

cout << "Введите площадь территории (в км^2): ";

cin >> country[current\_size].area;

cin.ignore();

cout << "Введите фамилию правителя страны: ";

cin.getline(country[current\_size].president, 50);

int government;

cout << "Выберите форму правления страны (0 - монархия, 1 - республика): ";

cin >> government;

country[current\_size].government = static\_cast<FormOfGovernment>(government);

cout << "Элемент структуры создан. " << endl;

current\_size++; // увеличиваем current\_size после заполнения всех полей нового элемента массива

}

else { // иначе выводим сообщение о переполнении

cout << "Структура переполнена!" << endl;

}

}

void outputInfo() {

for (int i = 0; i < current\_size; i++) { // поочередно выводим информацию о странах

cout << "\nСтрана №" << i + 1 << endl;

cout << "Название: " << country[i].name << endl;

cout << "Столица: " << country[i].capital << endl;

cout << "Численность населения: " << country[i].population << endl;

cout << "Площадь страны: " << country[i].area << endl;

cout << "Правитель страны: " << country[i].president << endl;

cout << "Форма правления страны: " << (country[i].government == MONARCHY ? "монархия" : "республика") << endl;

}

}

void searchInfo() {

char capital\_name[20]; // переменная для поиска по столице

cout << "По какой столице вы хотите найти информацию? ";

cin >> capital\_name;

bool found = false;

for (int i = 0; i < current\_size; i++) {

if (strcmp(capital\_name, country[i].capital) == 0) { // если введенная страна найдена, то выводим информацию о ней

cout << "Информация о стране по выбранной столице: " << capital\_name << endl;

cout << "Столица: " << country[i].capital << endl;

cout << "Численность населения: " << country[i].population << endl;

cout << "Площадь страны: " << country[i].area << endl;

cout << "Правитель страны: " << country[i].president << endl;

cout << "Форма правления: " << (country[i].government == MONARCHY ? "монархия" : "республика") << endl;

found = true;

break;

}

}

if (!found) {

cout << "Страна не найдена." << endl;

}

}

void deleteInfo() {

char capital\_name[20]; // переменная для поиска по столице

cout << "Введите столицу страны, которую хотите удалить из структуры: ";

cin >> capital\_name;

bool found = false;

for (int i = 0; i < current\_size; i++) {

if (strcmp(capital\_name, country[i].capital) == 0) { // если введенная страна найдена, то удаляем ее из структуры

found = true;

for (int j = i; j < current\_size - 1; j++) {

country[j] = country[j + 1];

}

current\_size--;

cout << "Структурная переменная удалена." << endl;

break;

}

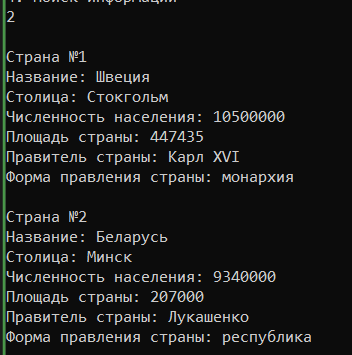
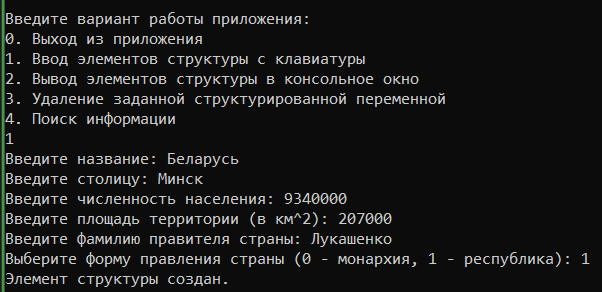
}

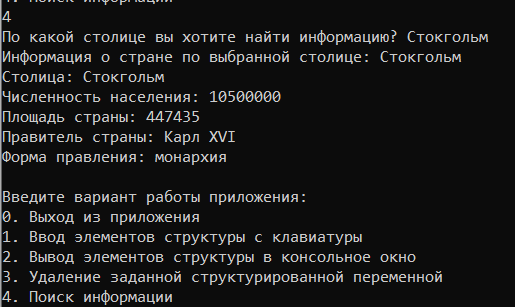
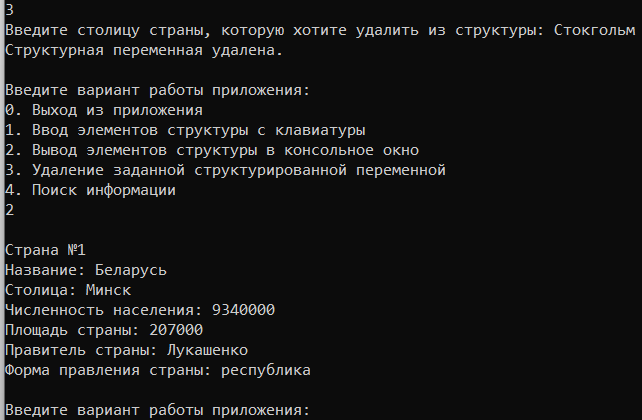
if (!found) {

cout << "Страна не найдена." << endl;

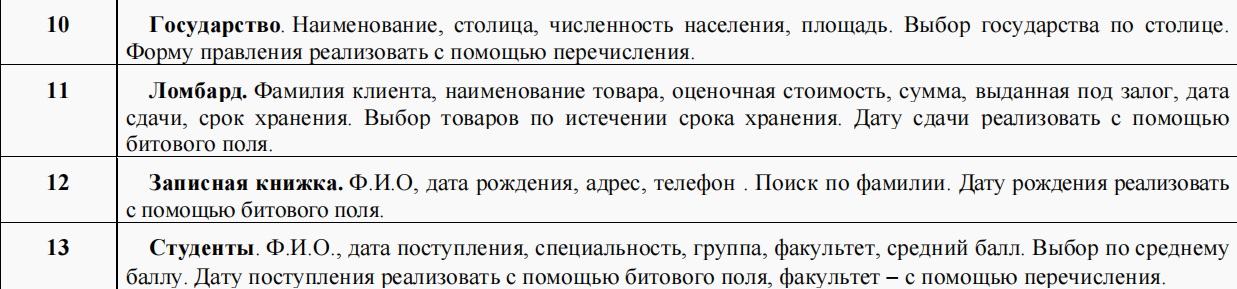
}

}





Вариант 11



#include <iostream>

#include <Windows.h>

#include <cstring>

using namespace std;

#define LIMIT 100 // максимальное количество стран для массива структуру

// структура

struct Pawnshop {

char last\_name[16];

char goods[40];

unsigned int assessed\_value;

unsigned int amount;

struct due\_date {

unsigned day : 5;

unsigned month : 4;

unsigned year : 12;

} hand\_over;

struct shelf {

unsigned day : 5;

unsigned month : 4;

unsigned year : 12;

} shelf\_life;

};

Pawnshop lombard[LIMIT]; // массив lombard

int current\_size = 0; // текущий размер массива

// прототипы функций

void inputInfo();

void outputInfo();

void searchInfo();

void deleteInfo();

// главная функция

int main() {

SetConsoleOutputCP(1251); // для распознавания кириллицы при записи в массив

SetConsoleCP(1251);

int choice;

do {

cout << "\nВведите вариант работы приложения: \n";

cout << "0. Выход из приложения\n";

cout << "1. Ввод элементов структуры с клавиатуры\n";

cout << "2. Вывод элементов структуры в консольное окно\n";

cout << "3. Удаление заданной структурированной переменной\n";

cout << "4. Поиск информации\n";

cin >> choice;

switch (choice) {

case 1: // Ввод элементов структуры с клавиатуры

inputInfo();

break;

case 2: // Вывод элементов структуры в консольное окно

outputInfo();

break;

case 3: // Удаление заданной структурированной переменной

deleteInfo();

break;

case 4: // Поиск информации

searchInfo();

break;

case 0: // Выход из программы

exit(0);

default:

cout << "Введено некорректное значение. " << endl;

break;

}

} while (choice != 0);

return 0;

}

// ввод информации

void inputInfo() {

if (current\_size < LIMIT) { // если массив не выходит за пределы заданного, то вводим соотв. информацию

cout << "Введите фамилию клиента: ";

cin >> lombard[current\_size].last\_name;

cin.ignore();

cout << "Введите наименование товара: ";

cin.getline(lombard[current\_size].goods, 50);

cout << "Введите оценочную стоимость товара: ";

cin >> lombard[current\_size].assessed\_value;

cout << "Введите сумму, выданную под залог: ";

cin >> lombard[current\_size].amount;

cout << "Введите дату сдачи в формате ДД ММ ГГГГ: ";

int day, month, year;

cin >> day >> month >> year;

lombard[current\_size].hand\_over.day = day;

lombard[current\_size].hand\_over.month = month;

lombard[current\_size].hand\_over.year = year;

cin.ignore();

cout << "Введите срок хранения товара в формате ДДММГГГГ: ";

cin >> day >> month >> year;

lombard[current\_size].shelf\_life.day = day;

lombard[current\_size].shelf\_life.month = month;

lombard[current\_size].shelf\_life.year = year;

cout << "Элемент структуры создан. " << endl;

current\_size++; // увеличиваем current\_size после заполнения всех полей нового элемента массива

}

else { // иначе выводим сообщение о переполнении

cout << "Структура переполнена!" << endl;

}

}

// вывод информации

void outputInfo() {

for (int i = 0; i < current\_size; i++) { // поочередно выводим информацию о странах

cout << "\nКлиент №" << i + 1 << endl;

cout << "Фамилия: " << lombard[i].last\_name << endl;

cout << "Товар: " << lombard[i].goods << endl;

cout << "Оценочная стоимость: " << lombard[i].assessed\_value << endl;

cout << "Сумма, выданную под залог: " << lombard[i].amount << endl;

cout << "Дата сдачи: " << lombard[i].hand\_over.day << "." << lombard[i].hand\_over.month << "." << lombard[i].hand\_over.year << endl;

cout << "Срок хранения: " << lombard[i].shelf\_life.day << "." << lombard[i].shelf\_life.month << "." << lombard[i].shelf\_life.year << endl;

}

}

// поиск просроченных товаров

void searchInfo() {

int cday, cmonth, cyear; //перменные для текущей даты

cout << "Введите текущую дату в формате ДД ММ ГГГГ для отсортировки просроченных товаров: ";

cin >> cday >> cmonth >> cyear;

bool found = false;

for (int i = 0; i < current\_size; i++) {

if (cyear > lombard[current\_size].shelf\_life.year ||

cmonth > lombard[current\_size].shelf\_life.month ||

(cmonth == lombard[current\_size].shelf\_life.month && cday > lombard[current\_size].shelf\_life.day))

{ // если просроченный товар найден, то удаляем его из структуры

cout << "Прсроченный товар найден: " << lombard[i].goods << endl;

cout << "Фамилия клиента: " << lombard[i].last\_name << endl;

cout << "Оценочная стоимость: " << lombard[i].assessed\_value << endl;

cout << "Сумма, выданную под залог: " << lombard[i].amount << endl;

cout << "Дата сдачи: " << lombard[i].hand\_over.day << "." << lombard[i].hand\_over.month << "." << lombard[i].hand\_over.year << endl;

cout << "Срок хранения: " << lombard[i].shelf\_life.day << "." << lombard[i].shelf\_life.month << "." << lombard[i].shelf\_life.year << endl;

found = true;

break;

}

}

if (!found) {

cout << "Просроченных товаров не найдено." << endl;

}

}

// удаление товаров из структуры

void deleteInfo() {

char item\_name[20]; // переменная для поиска товара

cout << "Введите название товара, который хотите удалить из структуры: ";

cin >> item\_name;

bool found = false; // по умолчанию считается, что товар не найден

for (int i = 0; i < current\_size; i++) {

if (strcmp(item\_name, lombard[i].goods) == 0) { // если введенный товар найден, то удаляем его из структуры

found = true;

for (int j = i; j < current\_size - 1; j++) {

lombard[j] = lombard[j + 1];

}

current\_size--;

cout << "Структурная переменная удалена." << endl;

break;

}

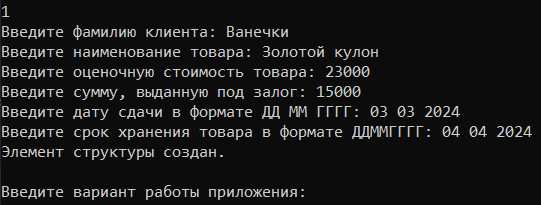
}

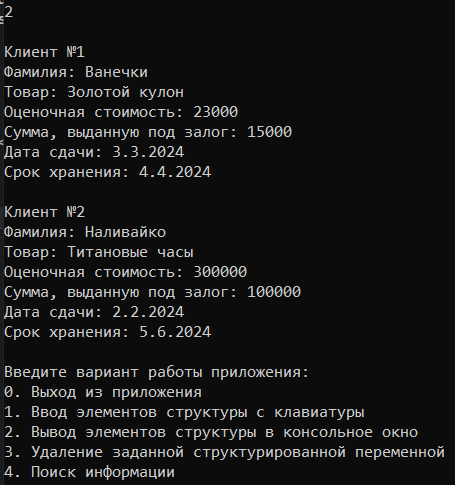
if (!found) {

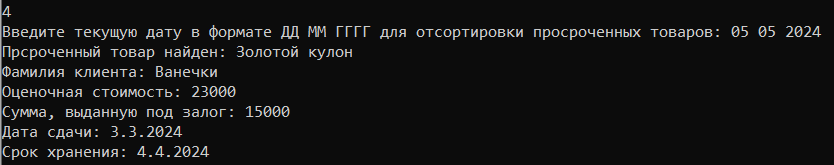
cout << "Товар не найден." << endl;

}

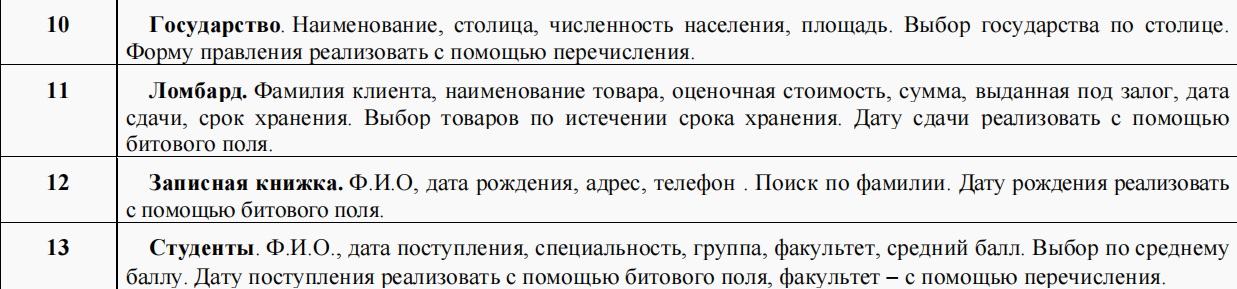
}







Вариант 13



#include <iostream>

#include <Windows.h>

#include <cstring>

using namespace std;

#define LIMIT 100 // максимальное количество стран для массива структуру

// структура

enum Faculties {

INFORMATION\_TECHNOLOGY = 0,

ENGINEERING\_ECONOMICS,

PRINTTECHNOLOGIES

};

struct Students {

char full\_name[50];

struct entrance\_date {

unsigned day : 5;

unsigned month : 4;

unsigned year : 12;

} entrance;

Faculties faculty;

char speciality[40];

short int group;

float average\_point;

};

Students university[LIMIT]; // массив university

int current\_size = 0; // текущий размер массива

// прототипы функций

void inputInfo();

void outputInfo();

void searchInfo();

void deleteInfo();

// главная функция

int main() {

SetConsoleOutputCP(1251); // для распознавания кириллицы при записи в массив

SetConsoleCP(1251);

int choice;

do {

cout << "\nВведите вариант работы приложения: \n";

cout << "0. Выход из приложения\n";

cout << "1. Ввод элементов структуры с клавиатуры\n";

cout << "2. Вывод элементов структуры в консольное окно\n";

cout << "3. Удаление заданной структурированной переменной\n";

cout << "4. Поиск информации\n";

cin >> choice;

switch (choice) {

case 1: // Ввод элементов структуры с клавиатуры

inputInfo();

break;

case 2: // Вывод элементов структуры в консольное окно

outputInfo();

break;

case 3: // Удаление заданной структурированной переменной

deleteInfo();

break;

case 4: // Поиск информации

searchInfo();

break;

case 0: // Выход из программы

exit(0);

default:

cout << "Введено некорректное значение. " << endl;

break;

}

} while (choice != 0);

return 0;

}

// ввод информации

void inputInfo() {

if (current\_size < LIMIT) { // если массив не выходит за пределы заданного, то вводим соотв. информацию

cin.ignore();

cout << "Введите ФИО студента: ";

cin.getline(university[current\_size].full\_name, 50);

cout << "Введите дату поступления в формате ДД ММ ГГГГ: ";

int day, month, year;

cin >> day >> month >> year;

university[current\_size].entrance.day = day;

university[current\_size].entrance.month = month;

university[current\_size].entrance.year = year;

int faculty;

cout << "Выберите факультет (0 - ФИТ, 1 - ИЭФ, 2 - ПИМ): ";

cin >> faculty;

university[current\_size].faculty = static\_cast<Faculties>(faculty);

cin.ignore();

cout << "Введите специальность: ";

cin.getline(university[current\_size].speciality, 50);

cout << "Введите номер группы: ";

cin >> university[current\_size].group;

cout << "Введите средний балл студента: ";

cin >> university[current\_size].average\_point;

cout << "Элемент структуры создан. " << endl;

current\_size++; // увеличиваем current\_size после заполнения всех полей нового элемента массива

}

else { // иначе выводим сообщение о переполнении

cout << "Структура переполнена!" << endl;

}

}

// вывод информации

void outputInfo() {

for (int i = 0; i < current\_size; i++) { // поочередно выводим информацию о странах

cout << "\nСтудент №" << i + 1 << endl;

cout << "ФИО: " << university[i].full\_name << endl;

cout << "Дата поступления: " << university[i].entrance.day << "." << university[i].entrance.month << "." << university[i].entrance.year << endl;

cout << "Факультет: ";

switch (university[i].faculty) {

case 0: cout << "Факультет информационных технологий" << endl; break;

case 1: cout << "Инженерно-экономический факультет" << endl; break;

case 2: cout << "Факультет принттехнологий и медиакоммуникаций" << endl; break;

default: cout << "не определился" << endl; break;

}

cout << "Специальность: " << university[i].speciality << endl;

cout << "Группа: " << university[i].group << endl;

cout << "Средний балл: " << university[i].average\_point << endl;

}

}

// поиск студентов по среднему баллу

void searchInfo() {

float GPA; // переменная для среднего балла студента

cout << "Введите средний балл студента, по которому хотите найти информацию подробнее: ";

cin >> GPA;

bool found = false;

for (int i = 0; i < current\_size; i++) {

if (GPA == university[i].average\_point) { // если такой студент найден, то выводим информацию о нем

found = true;

cout << "ФИО: " << university[i].full\_name << endl;

cout << "Дата поступления: " << university[i].entrance.day << "." << university[i].entrance.month << "." << university[i].entrance.year << endl;

cout << "Факультет: ";

switch (university[i].faculty) {

case 0: cout << "Факультет информационных технологий" << endl; break;

case 1: cout << "Инженерно-экономический факультет" << endl; break;

case 2: cout << "Факультет принттехнологий и медиакоммуникаций" << endl; break;

default: cout << "не определился" << endl; break;

}

cout << "Специальность: " << university[i].speciality << endl;

cout << "Группа: " << university[i].group << endl;

cout << "Средний балл: " << university[i].average\_point << endl;

break;

}

}

if (!found) {

cout << "Студентов с таким средним баллом не найдено." << endl;

}

}

// удаление студентов из структуры

void deleteInfo() {

float GPA; // переменная для среднего балла студента

cout << "Введите средний балл студента, которого хотите удалить его из структуры: ";

cin >> GPA;

bool found = false; // по умолчанию считается, что студент не найден

for (int i = 0; i < current\_size; i++) {

if (GPA == university[i].average\_point) { // если такой студент найден, то удаляем его из структуры

found = true;

for (int j = i; j < current\_size - 1; j++) {

university[j] = university[j + 1];

}

current\_size--;

cout << "Структурная переменная удалена." << endl;

break;

}

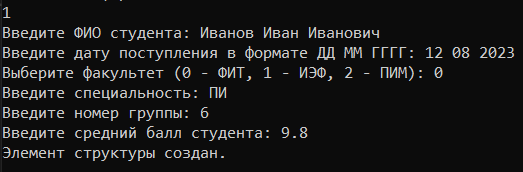
}

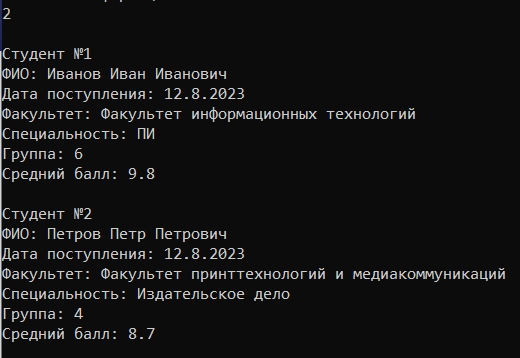
if (!found) {

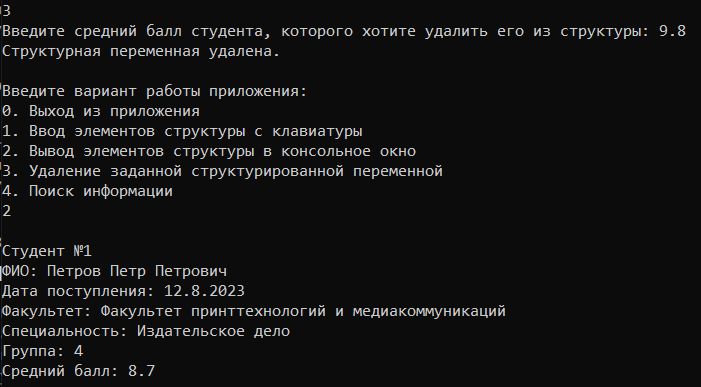
cout << "Студентов с таким средним баллом не найдено." << endl;

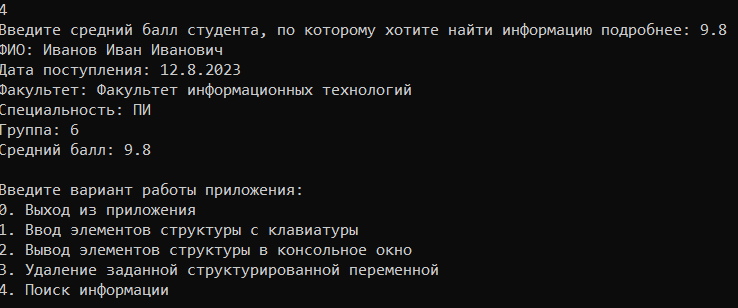
}

}

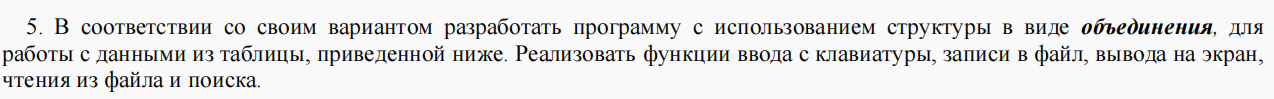




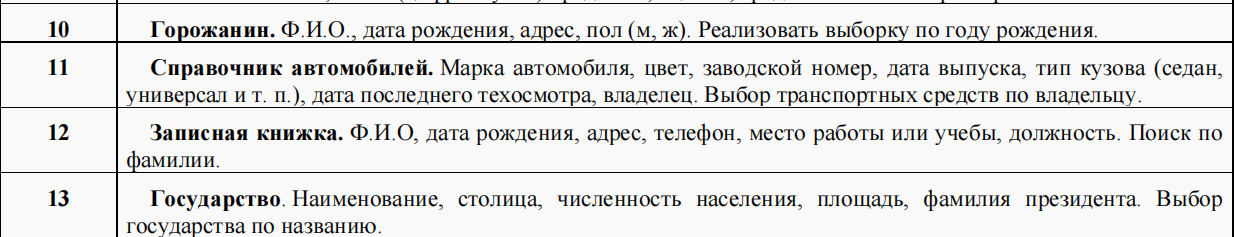




Задание 5



Вариант 12 (основной)



#include <iostream>

#include <Windows.h>

#include <fstream>

#define LIMIT 100 // максимальное количество элементов для массива структур

using namespace std;

// Структура записной книжки

struct Notebook {

char full\_name[50];

char birth\_date[16];

char address[40];

char phone\_number[15];

char workplace[40];

char position[40];

};

Notebook note[LIMIT]; // массив структур

int current\_size = 0; // текущий размер массива

// Прототипы функций

void inputInfo();

void outputInfo();

void searchInfo();

int main() {

SetConsoleOutputCP(1251); // для распознавания кириллицы при записи в массив

SetConsoleCP(1251);

int choice;

do {

cout << "\nВведите вариант работы приложения: \n";

cout << "0. Выход из приложения\n";

cout << "1. Ввод данных с клавиатуры и запись информации в файл\n";

cout << "2. Чтение информации из файла и вывод ее на экран и\n";

cout << "3. Поиск информации\n";

cin >> choice;

switch (choice) {

case 1: { // Ввод элементов структуры с клавиатуры и запись в файл

inputInfo();

break;

}

case 2: { // Чтение из файла и вывод элементов структуры в консольное окно

outputInfo();

break;

}

case 3: { // Поиск информации

searchInfo();

break;

}

case 0: { // Выход из программы

exit(0);

break;

}

default: {

cout << "Введено некорректное значение. " << endl;

break;

}

}

} while (choice != 0);

return 0;

}

void inputInfo() {

if (current\_size < LIMIT) { // Если массив не выходит за пределы заданного, то вводим соотв. информацию

cout << "Сколько записей хотите сделать? - ";

short int amount;

cin >> amount;

cin.ignore(); // Очистка буфера ввода

ofstream ofile("notes.txt"); // Открыть файл для добавления новых данных

if (!ofile) { // В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

// Ввод информации с клавиатуры и запись ее в файл

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

cout << "Запись №" << i + 1 << ": " << endl;

cout << "Введите ФИО: ";

cin.getline(note[current\_size].full\_name, 50);

ofile << note[current\_size].full\_name << ";";

cout << "Введите дату рождения: ";

cin.getline(note[current\_size].birth\_date, 16);

ofile << note[current\_size].birth\_date << ";";

cout << "Введите адрес: ";

cin.getline(note[current\_size].address, 40);

ofile << note[current\_size].address << ";";

cout << "Введите телефон: ";

cin.getline(note[current\_size].phone\_number, 15);

ofile << note[current\_size].phone\_number << ";";

cout << "Введите место работы или учебы: ";

cin.getline(note[current\_size].workplace, 40);

ofile << note[current\_size].workplace << ";";

cout << "Введите должность: ";

cin.getline(note[current\_size].position, 40);

ofile << note[current\_size].position << ";";

ofile << "\n";

current\_size++;

}

cout << "Информация успешно записана в файл" << endl;

ofile.close(); // Закрытие файла

}

else { // Иначе выводим сообщение о переполнении

cout << "Структура переполнена!" << endl;

}

}

void outputInfo() {

ifstream ifile("notes.txt"); // Создание объекта класса ifstream для чтения

if (!ifile) { // В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

int i = 0;

while (ifile.getline(note[i].full\_name, 50, ';')) {

cout << "\nЗапись №" << i + 1 << endl;

cout << "ФИО: " << note[i].full\_name << endl;

ifile.getline(note[i].birth\_date, 16, ';');

cout << "Дата рождения: " << note[i].birth\_date << endl;

ifile.getline(note[i].address, 40, ';');

cout << "Адрес: " << note[i].address << endl;

ifile.getline(note[i].phone\_number, 15, ';');

cout << "Телефон: " << note[i].phone\_number << endl;

ifile.getline(note[i].workplace, 40, ';');

cout << "Место работы или учебы: " << note[i].workplace << endl;

ifile.getline(note[i].position, 40, ';');

cout << "Должность: " << note[i].position << endl;

i++;

}

ifile.close();

}

void searchInfo() {

ifstream ifile("notes.txt"); // Создание объекта класса ifstream для чтения

if (!ifile) { // В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

char last\_name[20]; // Переменная для поиска по фамилии

cout << "По какой фамилии вы хотите найти информацию? " << endl;

cin >> last\_name;

bool found = false; // По умолчанию считается, что фамилия не найдена

while (ifile.getline(note[current\_size].full\_name, 50, ';')) {

if (strstr(note[current\_size].full\_name, last\_name) != nullptr) { // Если введенная фамилия найдена, то выводим информацию о ней

cout << "Информация по введенной фамилии: " << last\_name << endl;

ifile.getline(note[current\_size].birth\_date, 16, ';');

cout << "Дата рождения : " << note[current\_size].birth\_date << endl;

ifile.getline(note[current\_size].address, 40, ';');

cout << "Адрес: " << note[current\_size].address << endl;

ifile.getline(note[current\_size].phone\_number, 15, ';');

cout << "Телефон: " << note[current\_size].phone\_number << endl;

ifile.getline(note[current\_size].workplace, 40, ';');

cout << "Место учебы или должность: " << note[current\_size].workplace << endl;

ifile.getline(note[current\_size].position, 40, ';');

cout << "Должность: " << note[current\_size].position << endl;

found = true; // Устанавливаем, что фамилия найдена

break;

}

}

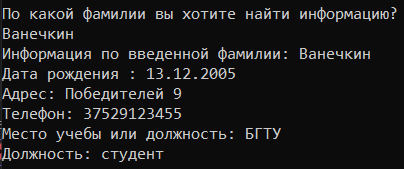
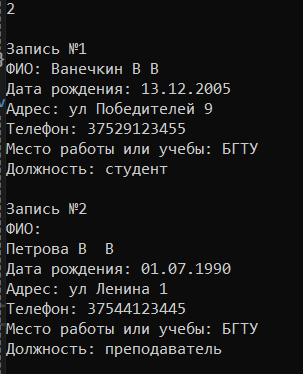
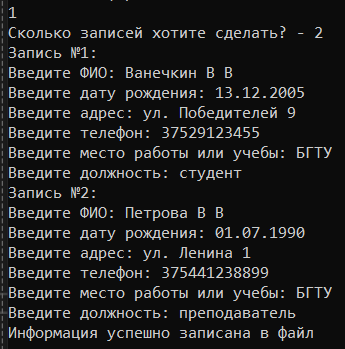
if (!found) { // Иначе выводим сообщение об ошибке

cout << "Фамилия была введена неверно" << endl;

}

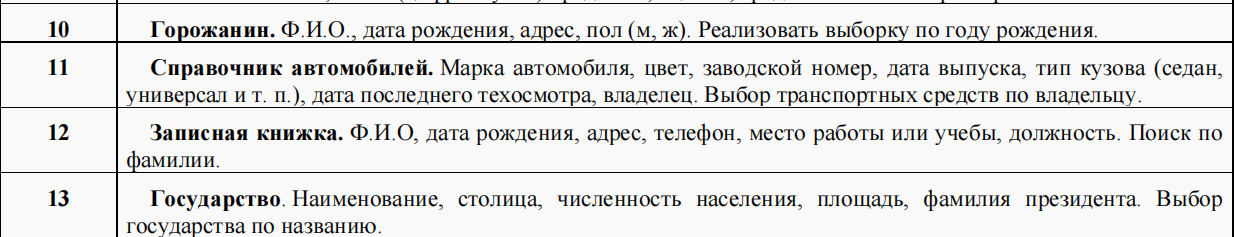
ifile.close();

}



Доп. задания

Вариант 10



#include <iostream>

#include <Windows.h>

#include <fstream>

#include <string>

#define LIMIT 100 // максимальное количество элементов для массива структур

using namespace std;

// объединение

union Notebook {

char full\_name[50];

char birth\_date[16];

char address[40];

char gender[2];

};

//структура

struct Records {

string full\_name;

string birth\_date;

string address;

string gender;

} records[LIMIT];

Notebook note[LIMIT]; // массив структур

int current\_size = 0; // текущий размер массива

// Прототипы функций

void inputInfo();

void outputInfo();

void searchInfo();

int main() {

SetConsoleOutputCP(1251); // для распознавания кириллицы при записи в массив

SetConsoleCP(1251);

int choice;

do {

cout << "\nВведите вариант работы приложения: \n";

cout << "0. Выход из приложения\n";

cout << "1. Ввод данных с клавиатуры и запись информации в файл\n";

cout << "2. Чтение информации из файла и вывод ее на экран и\n";

cout << "3. Поиск информации по году рождения\n";

cin >> choice;

switch (choice) {

case 1: { // Ввод элементов структуры с клавиатуры и запись в файл

inputInfo();

break;

}

case 2: { // Чтение из файла и вывод элементов структуры в консольное окно

outputInfo();

break;

}

case 3: { // Поиск информации по году рождения

searchInfo();

break;

}

case 0: { // Выход из программы

exit(0);

break;

}

default: {

cout << "Введено некорректное значение. " << endl;

break;

}

}

} while (choice != 0);

return 0;

}

void inputInfo() {

if (current\_size < LIMIT) { // Если массив не выходит за пределы заданного, то вводим соотв. информацию

cout << "Сколько записей хотите сделать? - ";

short int amount;

cin >> amount;

cin.ignore(); // Очистка буфера ввода

ofstream ofile("notes.txt"); // Открыть файл для добавления новых данных

if (!ofile) { // В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

// Ввод информации с клавиатуры и запись ее в файл

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

cout << "Запись №" << i + 1 << ": " << endl;

cout << "Введите ФИО: ";

cin.getline(note[i].full\_name, 50);

records[i].full\_name = note[i].full\_name;

ofile << note[i].full\_name << ";";

cout << "Введите дату рождения (формат ДД.ММ.ГГГГ): ";

cin.getline(note[i].birth\_date, 16);

records[i].birth\_date = note[i].birth\_date;

ofile << note[i].birth\_date << ";";

cout << "Введите адрес: ";

cin.getline(note[i].address, 40);

records[i].address = note[i].address;

ofile << note[i].address << ";";

cout << "Введите пол (м/ж): ";

cin >> note[i].gender;

records[i].gender = note[i].gender;

cin.ignore(); // Очистка буфера ввода

ofile << note[i].gender << endl;

current\_size++;

}

cout << "Информация успешно записана в файл" << endl;

ofile.close(); // Закрытие файла

}

else { // Иначе выводим сообщение о переполнении

cout << "Структура переполнена!" << endl;

}

}

void outputInfo() {

ifstream ifile("notes.txt"); // Создание объекта класса ifstream для чтения

if (!ifile) { // В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

int i = 0;

while (ifile.getline(note[i].full\_name, 50, ';')) {

cout << "\nЗапись №" << i + 1 << endl;

cout << "ФИО: " << note[i].full\_name << endl;

ifile.getline(note[i].birth\_date, 16, ';');

cout << "Дата рождения: " << note[i].birth\_date << endl;

ifile.getline(note[i].address, 40, ';');

cout << "Адрес: " << note[i].address << endl;

ifile >> note[i].gender;

cout << "Пол: " << note[i].gender << endl;

i++;

}

ifile.close();

}

void searchInfo() {

string year; // Переменная для поиска по году рождения

cout << "Введите год рождения для поиска: ";

cin >> year;

bool found = false; //по умолчанию считается, что год не найден

for (int i = 0; i < current\_size; i++) {

if (records[i].birth\_date.find(year) != string::npos) { // Если введенный год рождения найден, то выводим информацию о нем

cout << "\nИнформация по году рождения " << year << ":" << endl;

cout << "ФИО: " << records[i].full\_name << endl;

cout << "Дата рождения: " << records[i].birth\_date << endl;

cout << "Адрес: " << records[i].address << endl;

cout << "Пол: " << records[i].gender << endl;

found = true; // устанавливаем, что год рождения найден

}

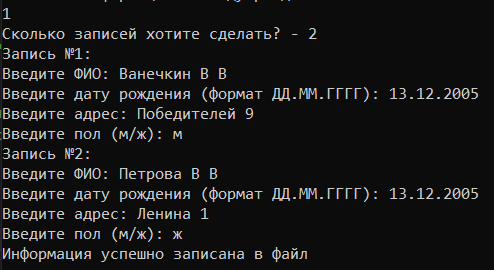
}

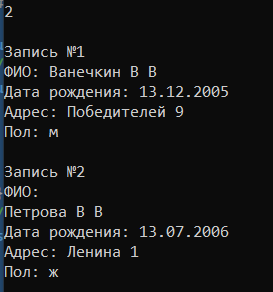
if (!found) { // Иначе выводим сообщение об ошибке

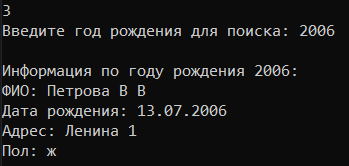
cout << "Нет информации о годе рождения " << year << endl;

}

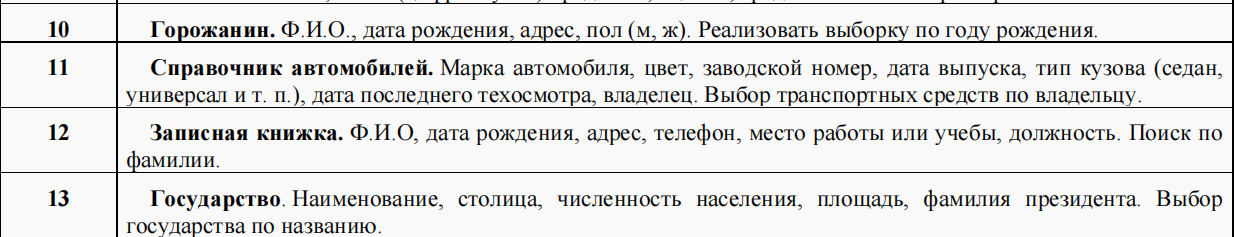
}







Вариант 11



#include <iostream>

#include <Windows.h>

#include <fstream>

#include <string>

#define LIMIT 100 // максимальное количество элементов для массива структур

using namespace std;

// объединение

union Ucars {

char brand[20];

char color[20];

char number[20];

char release\_date[20];

char body[20];

char last\_check[20];

char owner[20];

} note[LIMIT];

// структура

struct Cars {

string brand;

string color;

string number;

string release\_date;

string body;

string last\_check;

string owner;

} cars[LIMIT];

int current\_size = 0; // текущий размер массива

// Прототипы функций

void inputInfo();

void outputInfo();

void searchInfo();

int main() {

SetConsoleOutputCP(1251); // для распознавания кириллицы при записи в массив

SetConsoleCP(1251);

int choice;

do {

cout << "\nВведите вариант работы приложения: \n";

cout << "0. Выход из приложения\n";

cout << "1. Ввод данных с клавиатуры и запись информации в файл\n";

cout << "2. Чтение информации из файла и вывод ее на экран и\n";

cout << "3. Поиск информации по владельцу\n";

cin >> choice;

switch (choice) {

case 1: { // Ввод элементов структуры с клавиатуры и запись в файл

inputInfo();

break;

}

case 2: { // Чтение из файла и вывод элементов структуры в консольное окно

outputInfo();

break;

}

case 3: { // Поиск информации по владельцу

searchInfo();

break;

}

case 0: { // Выход из программы

exit(0);

break;

}

default: {

cout << "Введено некорректное значение. " << endl;

break;

}

}

} while (choice != 0);

return 0;

}

void inputInfo() {

if (current\_size < LIMIT) { // Если массив не выходит за пределы заданного, то вводим соотв. информацию

cout << "Сколько записей хотите сделать? - ";

short int amount;

cin >> amount;

cin.ignore(); // Очистка буфера ввода

ofstream ofile("cars.txt"); // Открыть файл для добавления новых данных

if (!ofile) { // В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

// Ввод информации с клавиатуры и запись ее в файл

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

cout << "Запись №" << i + 1 << ": " << endl;

cout << "Введите марку автомобиля: ";

cin.getline(note[i].brand, 20);

ofile << note[i].brand << ";";

cars[i].brand = note[i].brand;

cout << "Введите цвет: ";

cin.getline(note[i].color, 20);

ofile << note[i].color << ";";

cars[i].color = note[i].color;

cout << "Введите заводской номер: ";

cin.getline(note[i].number, 20);

ofile << note[i].number << ";";

cars[i].number = note[i].number;

cout << "Введите дату выпуска: ";

cin.getline(note[i].release\_date, 20);

ofile << note[i].release\_date << ";";

cars[i].release\_date = note[i].release\_date;

cout << "Введите тип кузова: ";

cin.getline(note[i].body, 20);

ofile << note[i].body << ";";

cars[i].body = note[i].body;

cout << "Введите последнего осмотра: ";

cin.getline(note[i].last\_check, 20);

ofile << note[i].last\_check << ";";

cars[i].last\_check = note[i].last\_check;

cout << "Введите владельца: ";

cin.getline(note[i].owner, 20);

cars[i].owner = note[i].owner;

ofile << note[i].last\_check << ";" << endl;

current\_size++;

}

cout << "Информация успешно записана в файл" << endl;

ofile.close(); // Закрытие файла

}

else { // Иначе выводим сообщение о переполнении

cout << "Структура переполнена!" << endl;

}

}

void outputInfo() {

ifstream ifile("cars.txt"); // Создание объекта класса ifstream для чтения

if (!ifile) { // В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

int i = 0;

while (ifile.getline(note[i].brand, 20, ';')) {

cout << "\nЗапись №" << i + 1 << endl;

cout << "Марка: " << note[i].brand << endl;

ifile.getline(note[i].color, 20, ';');

cout << "Цвет: " << note[i].color << endl;

ifile.getline(note[i].number, 20, ';');

cout << "Заводской номер: " << note[i].number << endl;

ifile.getline(note[i].release\_date, 20, ';');

cout << "Дата выпуска: " << note[i].release\_date << endl;

ifile.getline(note[i].body, 20, ';');

cout << "Тип кузова: " << note[i].body << endl;

ifile.getline(note[i].last\_check, 20, ';');

cout << "Дата последнего техосмотра: " << note[i].last\_check << endl;

ifile.getline(note[i].owner, 20, ';');

cout << "Владелец: " << note[i].owner << endl;

i++;

}

ifile.close();

}

void searchInfo() {

string owner; // Переменная для поиска по владельцу

cout << "Введите имя владельца для поиска: ";

cin >> owner;

bool found = false; //по умолчанию считается, что владелец не найден

for (int i = 0; i < current\_size; i++) {

if (cars[i].owner.find(owner) != string::npos) { // Если введенный владелец найден, то выводим информацию о нем

cout << "\nИнформация по владельцу " << owner << ":" << endl;

cout << "Марка: " << cars[i].brand << endl;

cout << "Цвет: " << cars[i].color << endl;

cout << "Заводской номер: " << cars[i].number << endl;

cout << "Дата выпуска: " << cars[i].release\_date << endl;

cout << "Тип кузова: " << cars[i].body << endl;

cout << "Дата последнего техосмотра: " << cars[i].last\_check << endl;

cout << "Владелец: " << cars[i].owner << endl;

found = true; // Устанавливаем, что владелец найден

}

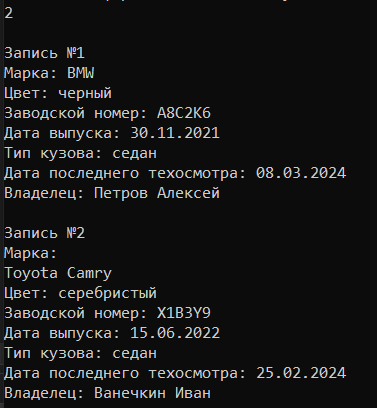
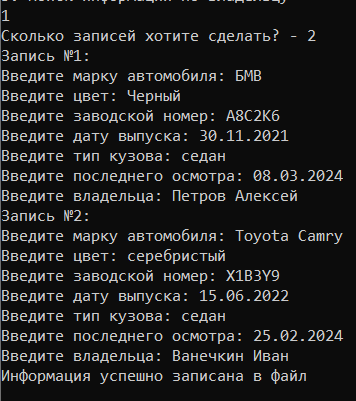
}

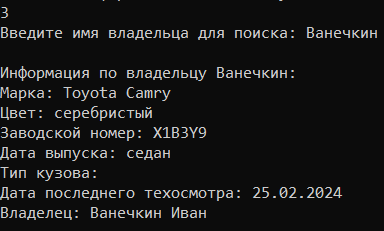
if (!found) { // Иначе выводим сообщение об ошибке

cout << "Нет информации о владельце " << owner << endl;

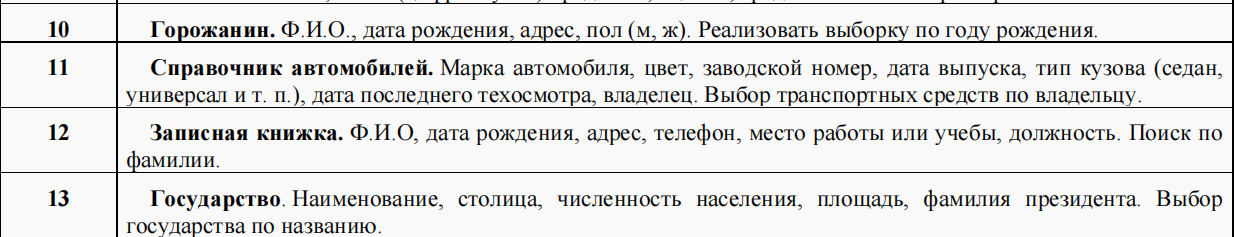
}

}





Вариант 13



#include <iostream>

#include <Windows.h>

#include <iostream>

#include <fstream>

#define LIMIT 100 // максимальное количество элементов для массива структур

//прототипы функций

using namespace std;

void inputInfo();

void outputInfo();

void searchInfo();

// структура

union Goverment

{

char name[16];

char capital[16];

unsigned int population;

unsigned int area;

char president[16];

};

Goverment country[LIMIT]; //массив country из объектов country

int current\_size = 0; // текущий размер массива

int main()

{

SetConsoleOutputCP(1251); // для распознавания кириллицы при записи в массив

SetConsoleCP(1251);

int choice;

do {

cout << "\nВведите вариант работы приложения: \n";

cout << "0. Выход из приложения\n";

cout << "1. Ввод данных с клавиатуры и запись информации в файл\n";

cout << "2. Чтение информации из файла и вывод ее на экран и\n";

cout << "3. Поиск информации\n";

cin >> choice;

switch (choice)

{

case 1: { //Ввод элементов структуры с клавиатуры и запись в файл

inputInfo();

break;

}

case 2: { //Чтение из файла и вывод элементов структуры в консольное окно

outputInfo();

break;

}

case 3: { //Поиск информации

searchInfo();

break;

}

case 0: { //Выход из программы

exit(0);

break;

}

default: {

cout << "Введено некорректное значение. " << endl;

break;

}

}

} while (choice != 0);

}

void inputInfo() {

if (current\_size < LIMIT) { //если массив не выходит за пределы заданного, то вводим соотв. информацию

cout << "Сколько стран хотите записать? - ";

short int amount;

cin >> amount;

ofstream ofile("countries.txt"); // открыть файл для добавления новых данных

if (ofile.fail()) { // в случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

//ввод информации с клавиатуры и запись ее в файл

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

cout << "Страна №" << i + 1 << ": " << endl;

cout << "Введите название: ";

cin >> country[current\_size].name;

ofile << country[current\_size].name << " ";

cout << "Введите столицу: ";

cin >> country[current\_size].capital;

ofile << country[current\_size].capital << " ";

cout << "Введите численность населения: ";

cin >> country[current\_size].population;

ofile << country[current\_size].population << " ";

cout << "Введите площадь территории (в км^2): ";

cin >> country[current\_size].area;

ofile << country[current\_size].area << " ";

cout << "Введите фамилию президента страны: ";

cin >> country[current\_size].president;

ofile << country[current\_size].president << " ";

ofile << "\n";

current\_size++;

}

cout << "Информация о странах успешно записана в файл" << endl;

ofile.close(); //закрытие файла

}

else { //иначе выводим сообщение о переполнении

cout << "Структура переполнена!" << endl;

}

}

void outputInfo() {

ifstream ifile("countries.txt"); // создание объекта класса ifstream для чтения

if (ifile.fail()) { // в случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

int i = 0;

while (ifile >> country[i].name) {

cout << "\nСтрана №" << i + 1 << endl;

cout << "Название: " << country[i].name << endl;

ifile >> country[i].capital;

cout << "Столица: " << country[i].capital << endl;

ifile >> country[i].population;

cout << "Численность населения: " << country[i].population << endl;

ifile >> country[i].area;

cout << "Площадь страны: " << country[i].area << endl;

ifile >> country[i].president;

cout << "Президент страны: " << country[i].president << endl;

i++;

}

ifile.close(); //закрытие файла

}

void searchInfo() {

ifstream ifile("countries.txt"); // создание объекста класса ifstream для чтения

if (ifile.fail()) { // в случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

char country\_name[20]; //переменная для поиска страны

cout << "По какой стране вы хотите найти информацию? " << endl;

cin >> country\_name;

bool found = false; // по умолчанию считается, что страна не найдена

while (ifile >> country[current\_size].name) {

ifile >> country[current\_size].name;

if (!strcmp(country\_name, country[current\_size].name)) { //если введенная страна найдена, то выводим информацию о ней

cout << "Информация по введенное стране: " << country\_name << endl;

ifile >> country[current\_size].capital; //запись информации из файла в объединение

cout << "Столица: " << country[current\_size].capital << endl; //запись информации из файла в объединение

ifile >> country[current\_size].population;

cout << "Численность населения: " << country[current\_size].population << endl;

ifile >> country[current\_size].area;

cout << "Площадь страны: " << country[current\_size].area << endl;

ifile >> country[current\_size].president;

cout << "Президент страны: " << country[current\_size].president << endl;

found = true; //устанавливаем, что страна найдена

break;

}

}

if (!found) { //иначе выводим сообщение об ошибке

cout << "Страна была введена неверно" << endl;

}

ifile.close();

}

