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:

Кулешов Артём

Студент 1 курса 8 группы

Преподаватель: асс. Андронова М.В.

Минск, 2024

4. В соответствии со своим вариантом разработать программу с использованием ***перечислений*** и ***битовых*** *полей* для работы с данными из таблицы, приведенной ниже. Реализовать функции ввода с клавиатуры, вывода на экран, удаления, поиска элементов. Интерфейс пользователя осуществить в виде меню.

Вариант №5

**Личная библиотека.** Автор книги, название, издательство, раздел библиотеки (специальная литература, хобби, домашнее хозяйство, беллетристика и т. д.), происхождение (покупка, кража, подарок) и наличие книги в данный момент. Выбор книг по году. Происхождение книги реализовать с помощью перечисления.

#include <iostream>

#include <string>

using namespace std;

//объявление функций//

void personal\_library();

void output();

void delete\_info();

void searching\_for();

int main()

{

int choice;

do {

cout << "Menu " << endl;

cout << "1. Enter information about book." << endl;

cout << "2. Information output." << endl;

cout << "3. Delete." << endl;

cout << "4. Searching for." << endl;

cout << "5. Exit." << endl;

cin >> choice;

switch (choice) {

case 1: personal\_library();

break;

case 2: output();

break;

case 3: delete\_info();

break;

case 4: searching\_for();

break;

case 5:

cout << "Exit." << endl;

break;

default:

cout << "Error" << endl;

break;

}

} while (choice != 5);

return 0;

}

//добавление перечисления//

enum origin

{

purchase = 1,

theft,

gift

};

//добавление структуры//

struct INFO

{

string author;

string title;

string publisher;

string year;

string library\_section;

origin origin;

string availability;

};

const int maxLibrary = 100;

int libraryCount = 1;

INFO library[maxLibrary];

//функция для доавления библиотеки//

void personal\_library()

{

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

cout << "Enter the author: ";

cin >> library[i].author;

cout << "Enter the title: ";

cin >> library[i].title;

cout << "Enter the publisher: ";

cin >> library[i].publisher;

cout << "Enter the year: ";

cin >> library[i].year;

cout << "Enter the library section (as like special literature, hobbies, home economics, fiction, etc.)";

cin >> library[i].library\_section;

int choice;

cout << "Enter origin (1 - purchase; 2 - theft; 3 - gift;): ";

cin >> choice;

switch (choice) {

case 1:

library[i].origin = purchase;

break;

case 2:

library[i].origin = theft;

break;

case 3:

library[i].origin = gift;

break;

default:

cout << "Error." << endl;

break;

}

cout << "Enter availability: ";

cin >> library[i].availability;

cout << "Adding successfully!" << endl;

}

}

//функция для вывода данных о библиотеке//

void output()

{

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

cout << "Library " << libraryCount << ": " << endl;

cout << "Author: ";

cout << library[i].author << endl;

cout << "Title: ";

cout << library[i].title << endl;

cout << "Publisher: ";

cout << library[i].publisher << endl;

cout << "Year: ";

cout << library[i].year << endl;

cout << "Library section: ";

cout << library[i].library\_section << endl;

cout << "Origin: ";

if (library[i].origin == purchase)

{

cout << "Purchase" << endl;

}

else if (library[i].origin == theft)

{

cout << "Theft" << endl;

}

else if (library[i].origin == gift)

{

cout << "Gift" << endl;

}

cout << "Availability: ";

cout << library[i].availability << endl;

}

}

//функция для удаление библиотеки

void delete\_info()

{

int delete\_info;

cout << endl << "Enter number of client: ";

cin >> delete\_info;

for (int i = delete\_info - 1; i < libraryCount - 1; i++)

{

library[i] = library[i + 1];

}

cout << endl << "Delete run success!" << endl;

libraryCount -= 1;

cout << endl;

}

//функция для поиска библиотеки по году//

void searching\_for()

{

string year;

cout << "Enter the year: ";

cin >> year;

bool found = false;

cout << "Results of the searching for " << year << ":" << endl;

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

if (library[i].year == year) {

found = true;

cout << "Author: " << library[i].author << endl;

cout << "Title: " << library[i].title << endl;

cout << "Publisher: " << library[i].publisher << endl;

cout << "Library section: " << library[i].library\_section << endl;

cout << "Origin: ";

if (library[i].origin == purchase)

{

cout << "Purchase" << endl;

}

else if (library[i].origin == theft)

{

cout << "Theft" << endl;

}

else if (library[i].origin == gift)

{

cout << "Gift" << endl;

}

cout << "Availability: " << library[i].availability << endl;

}

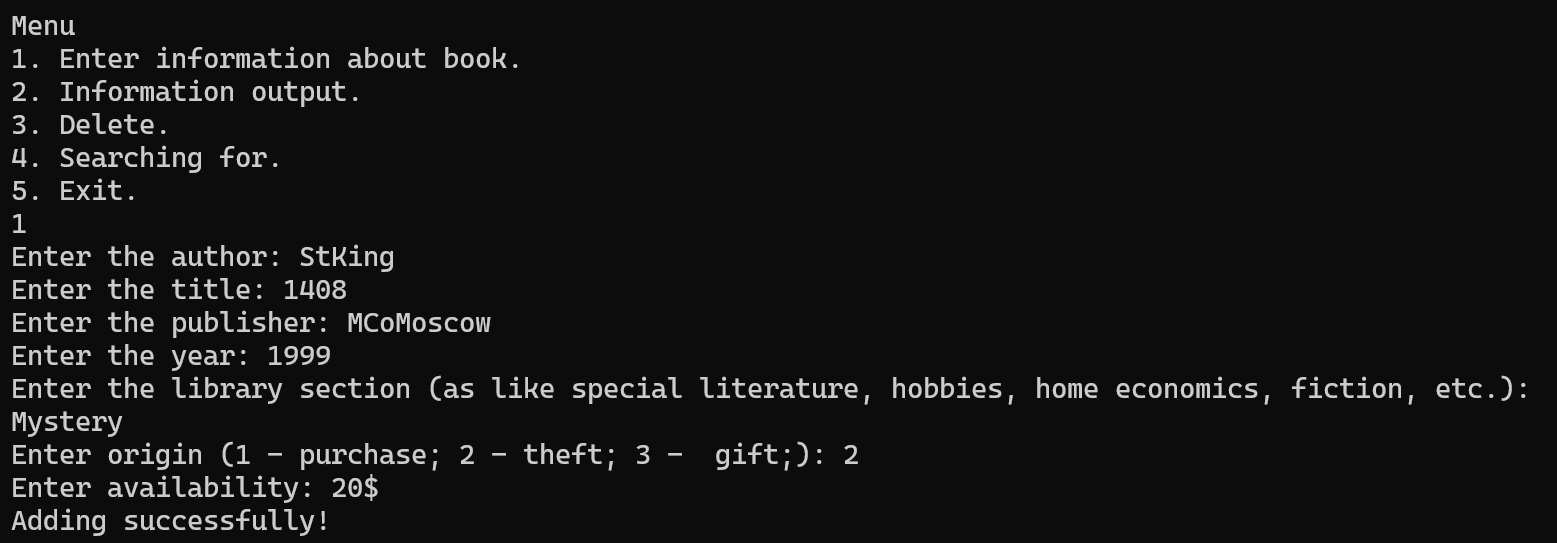
}

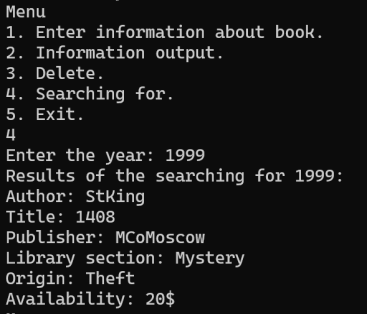
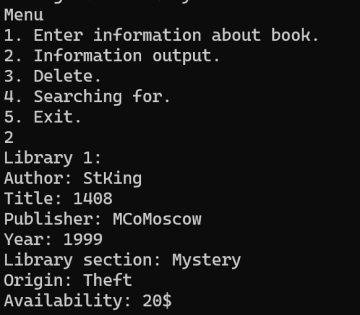
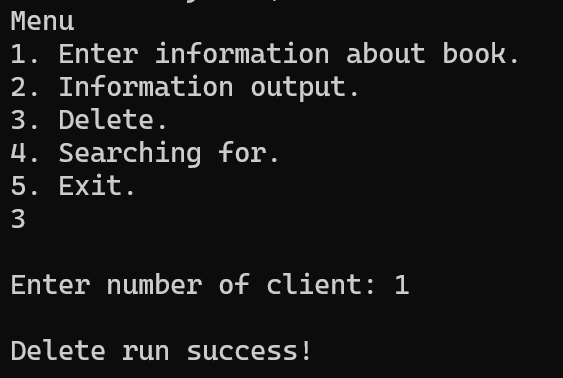
if (!found) {

cout << "Searching for " << year << " failed." << endl;

}

}



5. В соответствии со своим вариантом разработать программу с использованием структуры в виде ***объединения****,* для работы с данными из таблицы, приведенной ниже. Реализовать функции ввода с клавиатуры, записи в файл, вывода на экран, чтения из файла и поиска.

Вариант №5

**Личная библиотека.** Автор книги, название, издательство, раздел библиотеки (специальная литература, хобби, домашнее хозяйство, беллетристика и т. д.), происхождение (покупка, кража, подарок) и наличие книги в данный момент. Выбор книг по автору.

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

//объявление функций//

void personal\_library();

void output();

void delete\_info();

void searching\_for();

void write\_to\_file();

void read\_from\_file();

int main()

{

int choice;

do {

cout << "Menu " << endl;

cout << "1. Enter information about book." << endl;

cout << "2. Information output." << endl;

cout << "3. Delete." << endl;

cout << "4. Searching for." << endl;

cout << "5. Write to file." << endl;

cout << "6. Read from file." << endl;

cout << "7. Exit." << endl;

cin >> choice;

switch (choice) {

case 1: personal\_library();

break;

case 2: output();

break;

case 3: delete\_info();

break;

case 4: searching\_for();

break;

case 5:

write\_to\_file();

break;

case 6:

read\_from\_file();

break;

case 7:

cout << "Exit" << endl;

break;

default:

cout << "Error" << endl;

break;

}

} while (choice != 7);

return 0;

}

//добавление перечисления//

union Library

{

char author[10];

char title[7];

char publisher[15];

char year[4];

char library\_section[18];

char origin[100];

char availability[100];

};

//добавление структуры//

struct INFO

{

Library author;

Library title;

Library publisher;

Library year;

Library library\_section;

Library origin;

Library availability;

};

const int maxLibrary = 100;

int libraryCount = 1;

INFO library[maxLibrary];

//функция для доавления библиотеки//

void personal\_library()

{

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

cout << "Enter the author: ";

cin >> library[i].author.author;

cout << "Enter the title: ";

cin >> library[i].title.title;

cout << "Enter the publisher: ";

cin >> library[i].publisher.publisher;

cout << "Enter the year: ";

cin >> library[i].year.year;

cout << "Enter the library section (as like special literature, hobbies, home economics, fiction, etc.)" << endl;

cin >> library[i].library\_section.library\_section;

cout << "Enter origin (purchase; theft; gift;): ";

cin >> library[i].origin.origin;

cout << "Enter availability: ";

cin >> library[i].availability.availability;

cout << "Adding successfully!" << endl;

}

}

//функция для вывода данных о библиотеке//

void output()

{

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

cout << "Library " << libraryCount << ": " << endl;

cout << "Author: ";

cout << library[i].author.author << endl;

cout << "Title: ";

cout << library[i].title.title << endl;

cout << "Publisher: ";

cout << library[i].publisher.publisher << endl;

cout << "Year: ";

cout << library[i].year.year << endl;

cout << "Library section: ";

cout << library[i].library\_section.library\_section << endl;

cout << "Origin: ";

cout << library[i].origin.origin << endl;

cout << "Availability: ";

cout << library[i].availability.availability << endl;

}

}

//функция для удаление библиотеки

void delete\_info()

{

int delete\_info;

cout << endl << "Enter number of client: ";

cin >> delete\_info;

for (int i = delete\_info - 1; i < libraryCount - 1; i++)

{

library[i] = library[i + 1];

}

cout << endl << "Delete run success!" << endl;

libraryCount -= 1;

cout << endl;

}

//функция для поиска библиотеки по году//

void searching\_for()

{

string author;

cout << "Enter thr year: ";

cin >> author;

bool found = false;

cout << "Results of the searching for " << author << ":" << endl;

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

if (library[i].year.year == author) {

found = true;

cout << "Author: " << library[i].author.author << endl;

cout << "Title: " << library[i].title.title << endl;

cout << "Publisher: " << library[i].publisher.publisher << endl;

cout << "Library section: " << library[i].library\_section.library\_section << endl;

cout << "Origin: " << library[i].origin.origin << endl;

cout << "Availability: " << library[i].availability.availability << endl;

}

}

if (!found) {

cout << "Searching for " << author << " failed." << endl;

}

}

//функция для записи в файл//

void write\_to\_file()

{

ofstream write\_to\_file("file.txt");

if (!write\_to\_file.is\_open())

{

cout << "Error of opening file." << endl;

return;

}

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

{

//запись данных в файл//

write\_to\_file << "Library " << libraryCount << ": " << endl;

write\_to\_file << "Author: " << library[i].author.author << endl;

write\_to\_file << "Title: " << library[i].title.title << endl;

write\_to\_file << "Publisher: " << library[i].publisher.publisher << endl;

write\_to\_file << "Origin: " << library[i].origin.origin << endl;

write\_to\_file << "Year: " << library[i].year.year << endl;

write\_to\_file << "Library section: " << library[i].library\_section.library\_section << endl;

write\_to\_file << "Availability: " << library[i].availability.availability << endl << endl;

}

cout << "Writing run success!" << endl;

write\_to\_file.close();

}

//функция для чтения из файла//

void read\_from\_file()

{

ifstream read\_from\_file("file.txt");

string line;

while (getline(read\_from\_file, line))

{

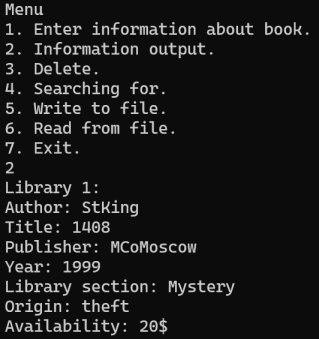
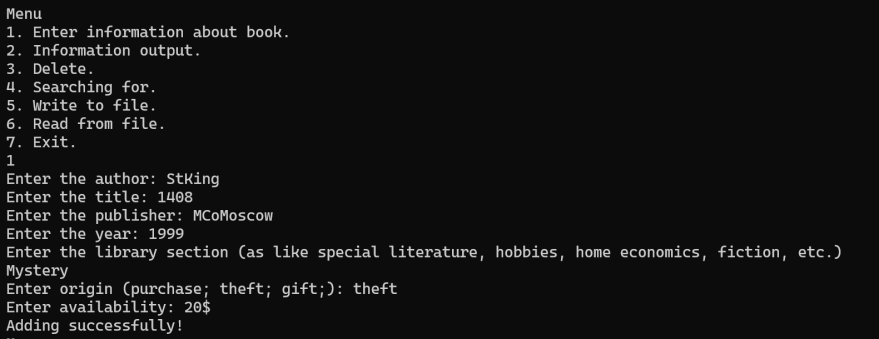
//вывод данных из файла//

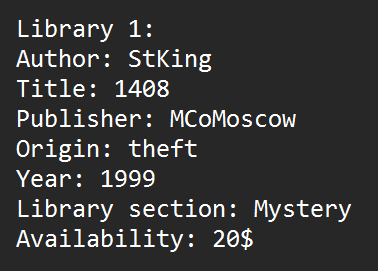
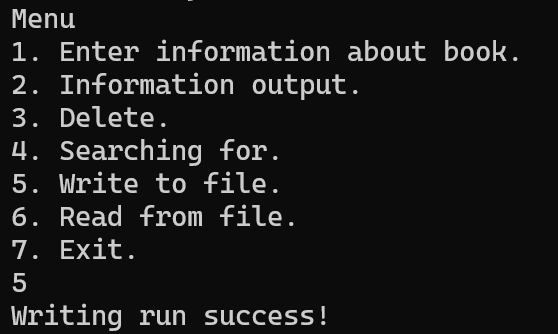
cout << line << endl;

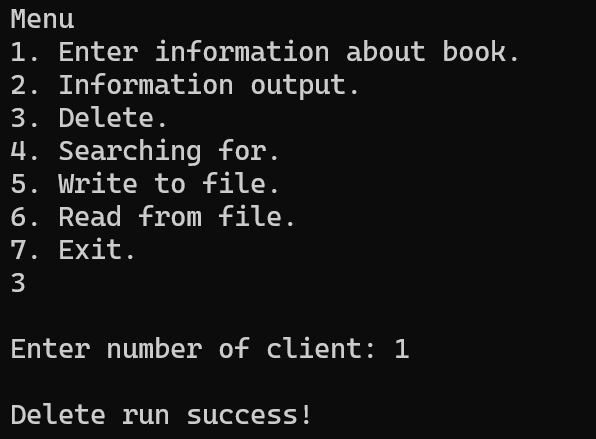
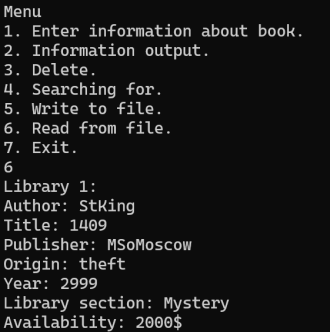
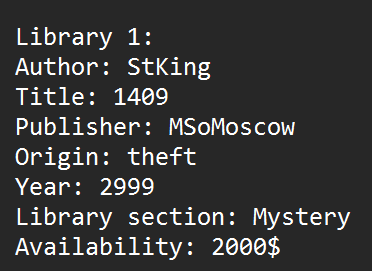
}

read\_from\_file.close();

}







Вариант №3

1. **Ученики**. Ф.И.О., класс (цифра+буква) предметы, оценки, средний балл. Выбор по фамилии. Класс реализовать с помощью битового поля, предметы − через перечисление.

#include <iostream>

#include <string>

#include <Windows.h>

using namespace std;

void add\_st();

void list\_of\_st();

void del\_st();

void search\_st();

const int MAX\_STUD\_COUNT = 5;

const int MAX\_LESSONS\_COUNT = 5;

const int MAX\_MARKS\_COUNT = 5;

int stud\_count = 0;

//перечисление для предметов

enum lessons

{

Algebra = 1,

Chemistry,

Physics,

Russian,

English

};

//структура для хранения данных об ученике

struct student

{

char fio[50];

unsigned studclass : 4;

char letter;

lessons lesson;

int marks[5];

double avg;

};

student stud\_list[MAX\_STUD\_COUNT];

int main()

{

int choice;

do

{

cout << "Menu " << endl;

cout << "1. Add new student." << endl;

cout << "2. Show all students." << endl

<< "3. Delete student" << endl

<< "4. Searching student." << endl

<< "5. Exit" << endl;

cin >> choice;

switch (choice)

{

case 1:

add\_st();

break;

case 2:

list\_of\_st();

break;

case 3:

del\_st();

break;

case 4:

search\_st();

break;

case 5:

cout << "Exit." << endl;

break;

}

} while (choice != 5);

}

//функция для добавления нового ученика

void add\_st()

{

student newStud;

cout << "Enter full name: ";

cin.ignore();

gets\_s(newStud.fio, 50);

cout << "Enter grad: ";

int cl;

cin >> cl;

newStud.studclass = static\_cast<unsigned>(cl);

cin.ignore();

cout << "Enter letter of grad:";

cin >> newStud.letter;

cin.ignore();

int choice;

cout << "Enter subject(1 - Algebra, 2 - Chemistry, 3 - Physics, 4 - Russian, 5 - English): ";

cin >> choice;

switch (choice)

{

case 1:

newStud.lesson = Algebra;

break;

case 2:

newStud.lesson = Chemistry;

break;

case 3:

newStud.lesson = Physics;

break;

case 4:

newStud.lesson = Russian;

break;

case 5:

newStud.lesson = English;

break;

default:

cout << "Error" << endl;

return;

break;

}

cout << "Enter students rating(max - 5): ";

for (int i = 0; i < MAX\_MARKS\_COUNT; i++)

{

cin >> newStud.marks[i];

}

int sum = 0;

for (int i = 0; i < MAX\_MARKS\_COUNT; i++)

{

sum += newStud.marks[i];

}

sum = static\_cast<double>(sum);

newStud.avg = sum / static\_cast<double>(MAX\_MARKS\_COUNT);

stud\_list[stud\_count] = newStud;

stud\_count++;

cout << "Data load success!" << endl << endl;

}

//функция для вывода списка учеников

void list\_of\_st()

{

for (int i = 0; i < stud\_count; i++)

{

cout << "Full name: " << stud\_list[i].fio << endl;

cout << "Grad: " << stud\_list[i].studclass << stud\_list[i].letter << endl;

cout << "Subject: ";

if (stud\_list[i].lesson == Algebra)

{

cout << "Algebra" << endl;

}

else if (stud\_list[i].lesson == Chemistry)

{

cout << "Chemistry" << endl;

}

else if (stud\_list[i].lesson == Physics)

{

cout << "Physics" << endl;

}

else if (stud\_list[i].lesson == Russian)

{

cout << "Russian" << endl;

}

else if (stud\_list[i].lesson == English)

{

cout << "English" << endl;

}

cout << "Rating: ";

for (int k = 0; k < MAX\_MARKS\_COUNT; k++)

{

cout << stud\_list[i].marks[k] << " ";

}

cout << endl;

cout << "Average rating: " << stud\_list[i].avg << endl << endl;

}

}

//функция для удаления ученика

void del\_st()

{

int choice;

cout << "Enter num of student: ";

cin >> choice;

for (int i = choice - 1; i < stud\_count - 1; i++)

{

stud\_list[i] = stud\_list[i + 1];

}

stud\_count--;

cout << "Data delete success!" << endl << endl;

}

//функция для поиска ученика по ФИО

void search\_st()

{

char buf[50];

bool found = false;

cout << "Enter full name: ";

cin.ignore();

gets\_s(buf, 50);

for (int i = 0; i < stud\_count; i++)

{

if (strcmp(buf, stud\_list[i].fio) == 0)

{

found = true;

cout << "Full name: " << stud\_list[i].fio << endl;

cout << "Grad: " << stud\_list[i].studclass << stud\_list[i].letter << endl;

cout << "Subject: ";

if (stud\_list[i].lesson == Algebra)

{

cout << "Algebra" << endl;

}

else if (stud\_list[i].lesson == Chemistry)

{

cout << "Биология" << endl;

}

else if (stud\_list[i].lesson == Physics)

{

cout << "Physics" << endl;

}

else if (stud\_list[i].lesson == Russian)

{

cout << "Russian" << endl;

}

else if (stud\_list[i].lesson == English)

{

cout << "English" << endl;

}

cout << "Rating: ";

for (int k = 0; k < MAX\_MARKS\_COUNT; k++)

{

cout << stud\_list[i].marks[k] << " ";

}

cout << endl;

cout << "Average rating: " << stud\_list[i].avg << endl << endl;

break;

}

}

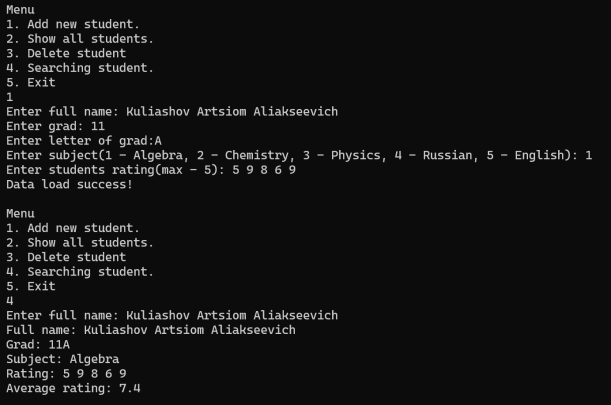
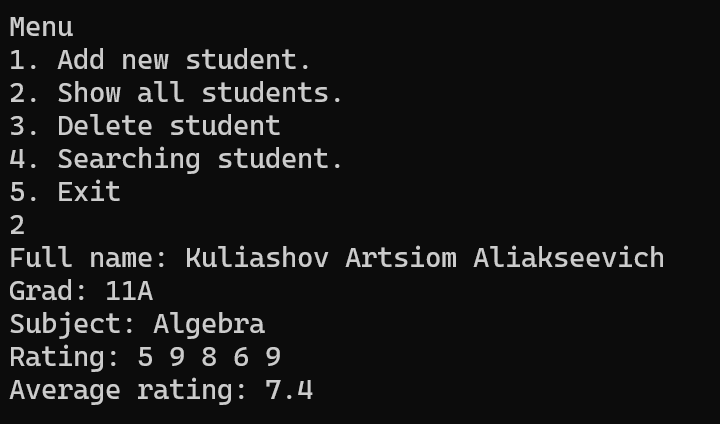
if (!found)

{

cout << "Error of finding student." << endl << endl;

}

}

**2. Ломбард.** База хранимых товаров и недвижимости: анкетные данные клиента, наименование товара, оценочная стоимость; сумма, выданная под залог, дата сдачи, срок хранения. Выбор товаров по наименованию.

#include <iostream>

#include <Windows.h>

#include <fstream>

#include <string>

using namespace std;

// Объявление структуры для хранения данных клиентов

struct pawnshop

{

    union buf

    {

        char ch[50];

        double db;

        int num;

    };

    buf fio;    // ФИО

    buf name;   // Наименование товара

    buf price;  // Оценочная стоимость товара

    buf sum;    // Сумма, выданная под залог

    buf date;   // Дата сдачи

    buf term;   // Срок хранения

};

// Константа для максимального количества клиентов

const int MAX\_CLIENTS\_COUNT = 5;

// Массив для хранения клиентов

pawnshop clients\_list[MAX\_CLIENTS\_COUNT];

// Переменная для отслеживания количества клиентов

int clients\_count = 0;

// Прототипы функций

void add();     // Добавить клиента

void list();    // Вывести данные клиентов

void write();   // Записать данные в файл

void read();    // Прочитать данные из файла

void search();  // Поиск по ФИО

// Основная функция программы

int main()

{

    // Установка кодировки консоли

    SetConsoleCP(1251);

    SetConsoleOutputCP(1251);

    int choice;

    // Главный цикл программы

    do

    {

        // Вывод меню

        cout << "1 - добавить клиента" << endl

            << "2 - вывести данные клиентов" << endl

            << "3 - записать данные в файл" << endl

            << "4 - прочитать данные из файла" << endl

            << "5 - поиск по ФИО" << endl

            << "6 - выход из программы" << endl

            << "Выберите пункт меню: ";

        // Ввод выбора пользователя

        cin >> choice;

        // Обработка выбора пользователя

        switch (choice)

        {

        case 1:

            add();

            break;

        case 2:

            list();

            break;

        case 3:

            write();

            break;

        case 4:

            read();

            break;

  case 5:

            search();

            break;

        case 6:

            cout << "Выход из программы..." << endl;

            system("pause");

            break;

        }

        }

    } while (choice != 6);

}

// Функция добавления клиента

void add()

{

    pawnshop newClient;

    // Проверка на максимальное количество клиентов

    if (clients\_count == MAX\_CLIENTS\_COUNT)

    {

        cout << "Вы ввели максимальное кол-во клиентов" << endl;

        return;

    }

    // Ввод данных о клиенте

    cout << "Введите ФИО клиента: ";

    cin.ignore();

    gets\_s(newClient.fio.ch, 50);

    cout << "Введите наименование товара: ";

    cin.ignore();

    gets\_s(newClient.name.ch, 50);

    cout << "Введите оценочную стоимость товара в $: ";

    cin >> newClient.price.db;

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

    cin >> newClient.sum.db;

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

    cin.ignore();

    gets\_s(newClient.date.ch, 50);

    cout << "Введите срок хранения в днях: ";

    cin >> newClient.term.num;

    // Добавление клиента в массив

    clients\_list[clients\_count] = newClient;

    // Увеличение счетчика клиентов

    clients\_count++;

    cout << "Данные успешно записаны!" << endl << endl;

}

// Функция вывода данных всех клиентов

void list()

{

    for (int i = 0; i < clients\_count; i++)

    {

        cout << "ФИО: " << clients\_list[i].fio.ch << endl;

        cout << "Наименование товара: " << clients\_list[i].name.ch << endl;

        cout << "Оценочная стоимость товара: " << clients\_list[i].price.db << "$" << endl;

        cout << "Сумма, выданная под залог: " << clients\_list[i].sum.db << "$" << endl;

        cout << "Дата сдачи: " << clients\_list[i].date.ch << endl;

        cout << "Срок хранения: " << clients\_list[i].term.num << " дней" << endl << endl;

    }

}

// Функция записи данных всех клиентов в файл

void write()

{

    ofstream fin("file.txt");

    // Проверка открытия файла

    if (!fin.is\_open())

    {

        cout << "Ошибка открытия файла" << endl << endl;

        return;

    }

    // Запись данных в файл

    for (int i = 0; i < clients\_count; i++)

    {

        fin << "ФИО: " << clients\_list[i].fio.ch << endl;

        fin << "Наименование товара: " << clients\_list[i].name.ch << endl;

        fin << "Оценочная стоимость товара: " << clients\_list[i].price.db << "$" << endl;

        fin << "Сумма, выданная под залог: " << clients\_list[i].sum.db << "$" << endl;

        fin << "Дата сдачи: " << clients\_list[i].date.ch << endl;

        fin << "Срок хранения: " << clients\_list[i].term.num << " дней" << endl << endl;

    }

    // Закрытие файла

    fin.close();

}

// Функция чтения данных из файла и вывода на экран

void read()

{

    ifstream fout("file.txt");

    string line;

    // Проверка открытия файла

    if (!fout.is\_open())

    {

        cout << "Ошибка открытия файла" << endl << endl;

        return;

    }

    // Чтение данных из файла и вывод на экран

    while (getline(fout, line))

    {

        cout << line << endl;

    }

    cout << endl;

    // Закрытие файла

    fout.close();

}

// Функция поиска клиента по ФИО

void search()

{

    char buffer[50];

    bool found = false;

    // Ввод ФИО для поиска

    cout << "Введите ФИО: ";

    cin.ignore();

    gets\_s(buffer, 50);

    // Поиск клиента по введенному ФИО

    for (int i = 0; i < clients\_count; i++)

    {

        if (strcmp(clients\_list[i].fio.ch, buffer) == 0)

        {

            found = true;

            cout << "ФИО: " << clients\_list[i].fio.ch << endl;

            cout << "Наименование товара: " << clients\_list[i].name.ch << endl;

            cout << "Оценочная стоимость товара: " << clients\_list[i].price.db << "$" << endl;

            cout << "Сумма, выданная под залог: " << clients\_list[i].sum.db << "$" << endl;

            cout << "Дата сдачи: " << clients\_list[i].date.ch << endl;

            cout << "Срок хранения: " << clients\_list[i].term.num << " дней" << endl << endl;

        }

    }

    // Вывод сообщения, если клиент не найден

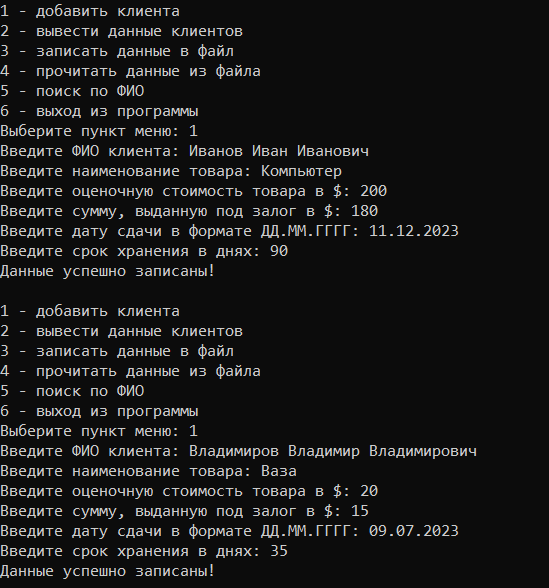
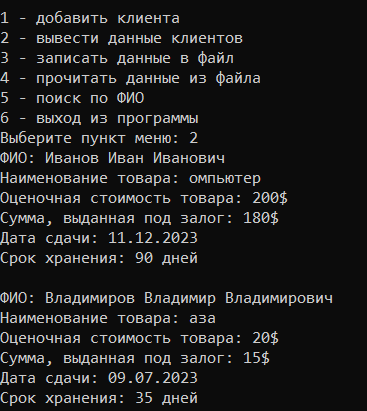
    if (!found)

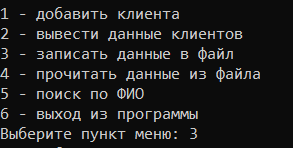
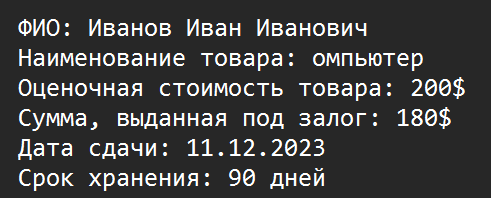
    {

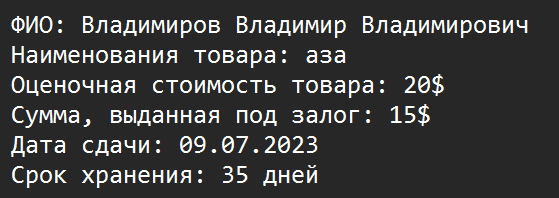
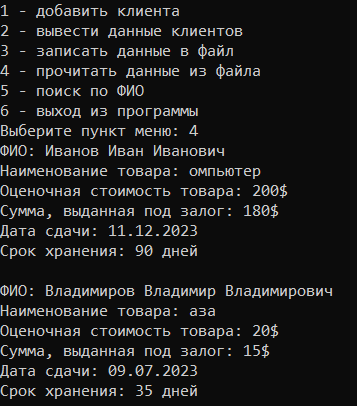
        cout << "Клиент с такими ФИО не найден" << endl << endl;

    }

}

** **

**** 

****

Вариант №12

**Записная книжка.** Ф.И.О, дата рождения, адрес, телефон . Поиск по фамилии. Дату рождения реализовать с помощью битового поля.

#include<iostream>

#include<string>

#include<windows.h>

#definesize 10 //подключение макроса для длины массива

usingnamespace std;

void input(); //прототип функции ввода

void output(); // прототип функции вывода

void del(); // прототип функции для удаления

void find(); // прототип функции поиска

structnotebook// структура

{

string name;

unsigned day : 5;

unsigned month : 4;

unsigned year : 14;

string addr;

string phonenumber;

};

notebook diary[size];

int choice;

int current\_size = 0;

int main()

{

setlocale(LC\_ALL, "Rus");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

//меню выбора

do

{

cout <<"1 - Ввод элементов с клавиатуры"<< endl;

cout <<"2 - Вывод элементов в консольное окно"<< endl;

cout <<"3 - Удаление заданной структурированной переменной"<< endl;

cout <<"4 - Поиск информации "<< endl;

cout <<"5 - Выход из программы"<< endl;

cin >> choice;

// анализируем choice

switch (choice)

{

case 1:

input();

break;

case 2:

output();

break;

case 3:

del();

break;

case 4:

find();

break;

}

} while (choice != 5);

return 0;

}

void input() // функция ввода

{

int day, month, year;

int numb; // для хранения количества

cout <<"Введите количество записей: "; cin >> numb;

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

{

if (current\_size <size) // условие для проверки наличия места

{

// заполнение информации

cout <<"Информация о "<< i + 1 <<" записи: "<< endl;

cin.ignore();

cout << endl <<"ФИО: "<< endl;

getline(cin, diary[current\_size].name);

cout <<"Адрес: "<< endl;

getline(cin, diary[current\_size].addr);

cout <<"Номер телефона: "<< endl;

getline(cin, diary[current\_size].phonenumber);

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

cin >> day >> month >> year;

diary[current\_size].day = day;

diary[current\_size].month = month;

diary[current\_size].year = year;

current\_size++;

}

}

}

void output() // функция вывода

{

// вывод информации

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

{

cout <<"Информация о "<< i + 1 <<" человеке в записной книжке: "<< endl;

cout << endl <<"ФИО: "<< diary[i].name << endl;

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

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

cout <<"Адрес: "<< diary[i].addr << endl;

cout <<"Номер телефона: "<< diary[i].phonenumber << endl;

}

}

void del() // функция для удаления

{

int d;

bool f = false;

string line;

cout <<"Введите ФИО для удаления информации"<< endl;

cin.ignore();

getline(cin, line);

for (int i = 0;i < current\_size;i++) // ищем подходящее

{

if (line==diary[i].name) // если нашли

{

d = i; // запоминаем

f = true;

}

}

if (f) // если нашли

{

for (int de1 = d; de1 < current\_size; de1++) // цикл для удаления

{

diary[de1] = diary[de1 + 1];

}

current\_size = current\_size - 1;

cout <<"Информация удалена"<< endl;

}

else

cout <<"Информация не найдена!"<< endl; // иначе выводим что не нашли

}

void find() // функция поиска

{

bool f = true;

string line;

cout <<"Введите ФМО для удаления информации"<< endl;

cin.ignore();

getline(cin, line);

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

{

if (line == diary[i].name) // если нашли выводим

{

cout <<"Информация о "<< i + 1 <<" записи: "<< endl;

cout << endl <<"ФИО: "<< diary[i].name << endl;

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

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

cout <<"Адрес: "<< diary[i].addr << endl;

cout <<"Номер телефона: "<< diary[i].phonenumber << endl;

f = false;

}

}

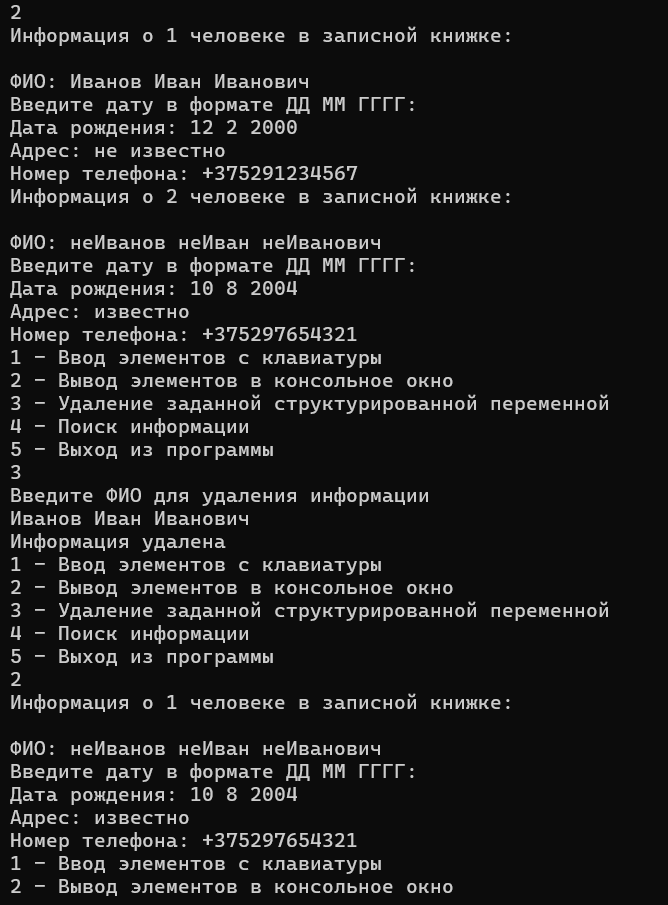
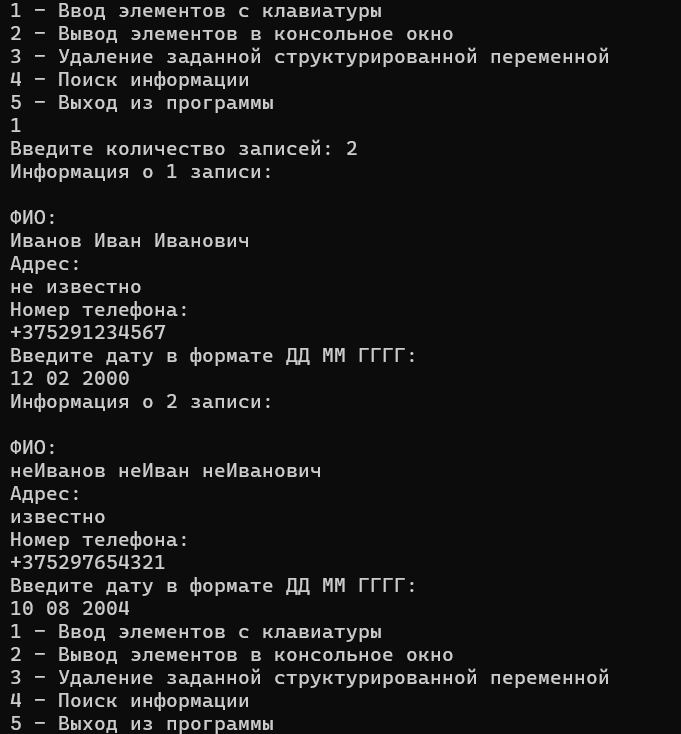
if (f)

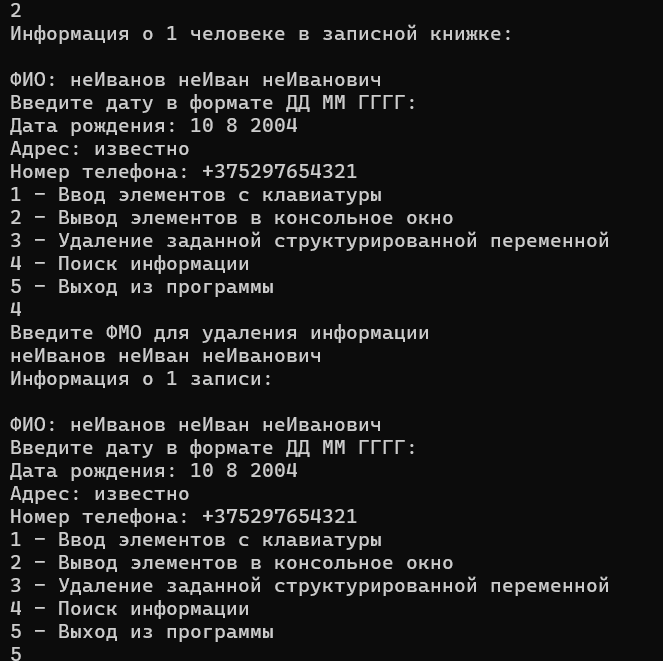
{

cout <<"Информация не найдена!"<< endl;

}

}





**Записная книжка.** Ф.И.О, дата рождения, адрес, телефон, место работы или учебы, должность. Поиск по фамилии.

#include<iostream>

#include<fstream>

#include<string>

#include<windows.h

#definesize 10 //подключение макроса для длины массива

usingnamespace std;

void input(); //прототип функции ввода

void output(); // прототип функции вывода

void inFile(); // прототип функции для записи в файл

void outFile(); // прототип функции для чтения из файла

void find(); // прототип функции поиска

unionbuf// объединение

{

// аналогичная ситуация

char name[50];

char date[50];

char addr[50];

char phonenumber[50];

char job[50];

char position[50];

};

structnotebook// структура

{

buf fullname;

buf birthday;

buf add;

buf phone;

buf pljob;

buf pos;

};

notebook diary[size];

int choice;

int current\_size;

int main()

{

setlocale(LC\_ALL, "Rus");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

//меню выбора

do

{

cout <<"1 - Ввод элементов с клавиатуры"<< endl;

cout <<"2 - Вывод элементов в консольное окно"<< endl;

cout <<"3 - Запись информации в файл"<< endl;

cout <<"4 - Чтение информации из файла"<< endl;

cout <<"5 - Поиск информации"<< endl;

cout <<"6 - Выход из программы"<< endl;

cin >> choice;

// анализируем choice

switch (choice)

{

case 1:

input();

break;

case 2:

output();

break;

case 3:

inFile();

break;

case 4:

outFile();

break;

case 5:

find();

break;

}

} while (choice != 6);

return 0;

}

void input() // функция ввода

{

int numb; // для хранения количества

int choice; // выбор из перечесления

cout <<"Введите количество записей: "; cin >> numb;

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

{

if (current\_size <size) // условие для проверки наличия места

{

// заполнение информации

cin.ignore();

cout <<"Информация о "<< i + 1 <<" человеке: "<< endl;

cout << endl <<"ФИО: "<< endl;

gets\_s(diary[current\_size].fullname.name, 50);

cout <<"Дата рождения: "<< endl;

gets\_s(diary[current\_size].birthday.date, 50);

cout <<"Адрес: "<< endl;

gets\_s(diary[current\_size].add.addr, 50);

cout <<"Телефон: "<< endl;

gets\_s(diary[current\_size].phone.phonenumber, 50);

cout <<"Место работы или учёбы: "<< endl;

gets\_s(diary[current\_size].pljob.job, 50);

cout <<"Должность: "<< endl;

gets\_s(diary[current\_size].pos.position, 50);

current\_size++;

}

}

}

void output() // функция вывода

{

// вывод информации

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

{

cout <<"Информация о "<< i + 1 <<" человеке: "<< endl;

cout << endl <<"ФИО: "<< diary[i].fullname.name << endl;

cout <<"Дата рождения: "<< diary[i].birthday.date << endl;

cout <<"Адрес: "<< diary[i].add.addr << endl;

cout <<"Телефон: "<< diary[i].phone.phonenumber << endl;

cout <<"Место работы или учёбы: "<< diary[i].pljob.job << endl;

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

}

}

void inFile() // функция записи в файл

{

ofstream file1("FILE1.txt");

if (file1)

{

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

{

file1 <<"Информация о "<< i + 1 <<" человеке: "<< endl;

file1 << endl <<"ФИО: "<< diary[i].fullname.name << endl;

file1 <<"Дата рождения: "<< diary[i].birthday.date << endl;

file1 <<"Адрес: "<< diary[i].add.addr << endl;

file1 <<"Телефон: "<< diary[i].phone.phonenumber << endl;

file1 <<"Место работы или учёбы: "<< diary[i].pljob.job << endl;

file1 <<"Должность: "<< diary[i].pos.position << endl;

}

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

file1.close();

}

else

cout <<"Не удалось открыть файл!"<< endl;

}

void outFile() // функция чтения из файла

{

ifstream file1("FILE1.txt");

string line;

while (getline(file1, line))

{

cout << line << endl;

}

file1.close();

}

void find() // функция поиска

{

string line;

bool f = true;

cout <<"Введите ФИО: ";

cin.ignore();

getline(cin, line);

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

{

if (line == diary[i].fullname.name)

{

cout <<"Информация о "<< i + 1 <<" человеке: "<< endl;

cout << endl <<"ФИО: "<< diary[i].fullname.name << endl;

cout <<"Дата рождения: "<< diary[i].birthday.date << endl;

cout <<"Адрес: "<< diary[i].add.addr << endl;

cout <<"Телефон: "<< diary[i].phone.phonenumber << endl;

cout <<"Место работы или учёбы: "<< diary[i].pljob.job << endl;

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

f = false;

}

}

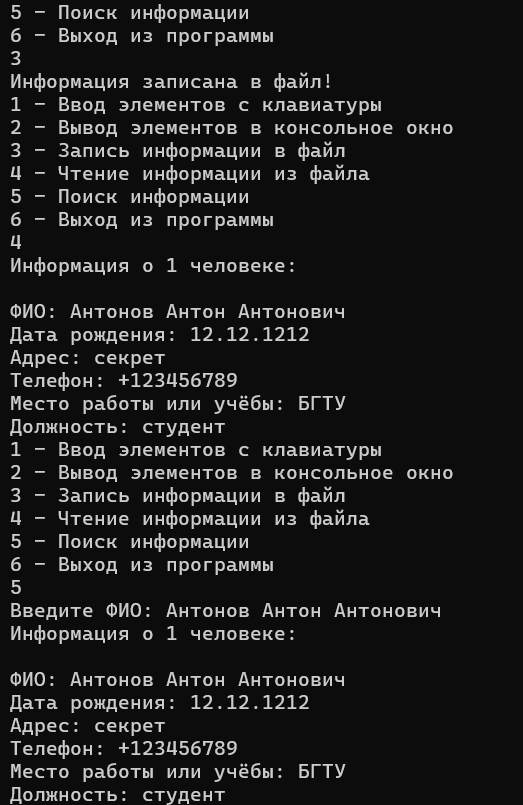
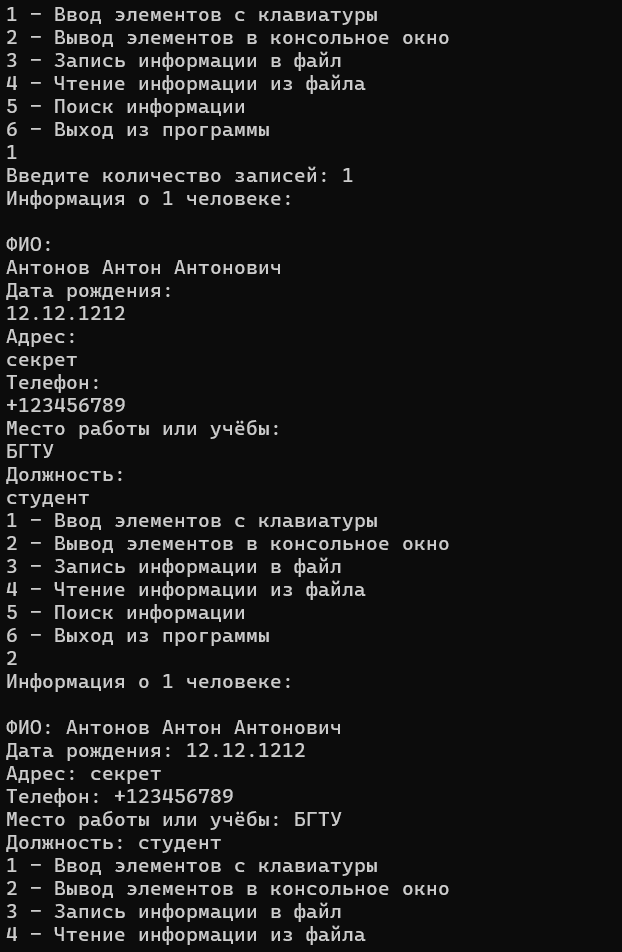
if (f)

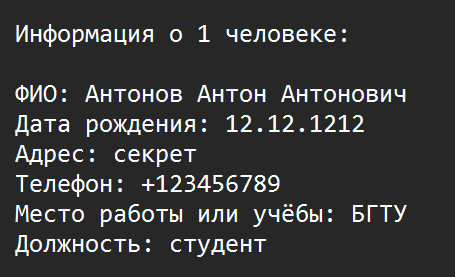
{

cout <<"Информация не найдена!"<< endl;

}

}





Вариант №13

**Студенты**. Ф.И.О., дата поступления, специальность, группа, факультет, средний балл. Выбор по среднему баллу. Дату поступления реализовать с помощью битового поля, факультет − с помощью перечисления.

#include<iostream>

#include<string>

#include<windows.h>

#definesize 10 //подключение макроса для длины массива

usingnamespace std;

void input(); //прототип функции ввода

void output(); // прототип функции вывода

void del(); // прототип функции для удаления

void find(); // прототип функции поиска

enumfaculty// перечисление

{

ФИТ = 1, не\_ФИТ

};

structstudent// структура

{

char name[50];

char speciality[50];

int group;

float average;

faculty fac;

unsigned day : 5;

unsigned month : 4;

unsigned year : 14;

};

student students[size];

int choice;

int current\_size = 0;

int main()

{

setlocale(LC\_ALL, "Rus");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

//меню выбора

do

{

cout <<"1 - Ввод элементов с клавиатуры"<< endl;

cout <<"2 - Вывод элементов в консольное окно"<< endl;

cout <<"3 - Удаление заданной структурированной переменной"<< endl;

cout <<"4 - Поиск информации "<< endl;

cout <<"5 - Выход из программы"<< endl;

cin >> choice;

// анализируем choice

switch (choice)

{

case 1:

input();

break;

case 2:

output();

break;

case 3:

del();

break;

case 4:

find();

break;

}

} while (choice != 5);

return 0;

}

void input() // функция ввода

{

int day, month, year;

int numb; // для хранения количества

cout <<"Введите количество записей: "; cin >> numb;

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

{

if (current\_size <size) // условие для проверки наличия места

{

// заполнение информации

cout <<"Информация о "<< i + 1 <<" студенте: "<< endl;

cin.ignore();

cout << endl <<"ФИО: "<< endl;

gets\_s(students[current\_size].name,50);

cout <<"Специальность: "<<endl;

gets\_s(students[current\_size].speciality, 50);

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

cin >> day >> month >> year;

students[current\_size].day = day;

students[current\_size].month = month;

students[current\_size].year = year;

cout <<"Факультет (1 - ФИТ, 2 - не ФИТ): "<< endl;

cin >> choice;

switch (choice)

{

case 1:

students[current\_size].fac = ФИТ;

break;

case 2:

students[current\_size].fac = не\_ФИТ;

break;

}

cout <<"Группа: "<< endl;

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

cout <<"Средний балл: "<< endl;

cin >> students[current\_size].average;

current\_size++;

}

}

}

void output() // функция вывода

{

// вывод информации

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

{

cout << endl <<"ФИО: "<< students[i].name << endl;

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

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

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

cout <<"Факультет: ";

if (students[i].fac == ФИТ)

{

cout <<"ФИТ";

}

else

if (students[i].fac == не\_ФИТ)

{

cout <<" не ФИТ";

}

cout << endl;

cout <<"Средний балл: "<< students[i].average << endl;

}

}

void del() // функция для удаления

{

int d;

bool f = false;

float av;

cout <<"Введите средний балл для удаления информации"<< endl;

cin >> av;

for (int i = 0;i < current\_size;i++) // ищем подходящее

{

if (av == students[i].average) // если нашли

{

d = i; // запоминаем

f = true;

}

}

if (f) // если нашли

{

for (int de1 = d; de1 < current\_size; de1++) // цикл для удаления

{

students[de1] = students[de1 + 1];

}

current\_size = current\_size - 1;

cout <<"Информация удалена"<< endl;

}

else

cout <<"Информация не найдена!"<< endl; // иначе выводим что не нашли

}

void find() // функция поиска

{

bool f = true;

float av;

cout <<"Введите средний балл для поиска информации: "<< endl;

cin >> av;

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

{

if (av == students[i].average) // если нашли выводим

{

cout << endl <<"ФИО: "<< students[i].name << endl;

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

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

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

cout <<"Факультет: ";

if (students[i].fac == ФИТ)

{

cout <<"ФИТ";

}

else

if (students[i].fac == не\_ФИТ)

{

cout <<"не ФИТ";

}

cout << endl;

cout <<"Средний балл: "<< students[i].average << endl;

f = false;

}

}

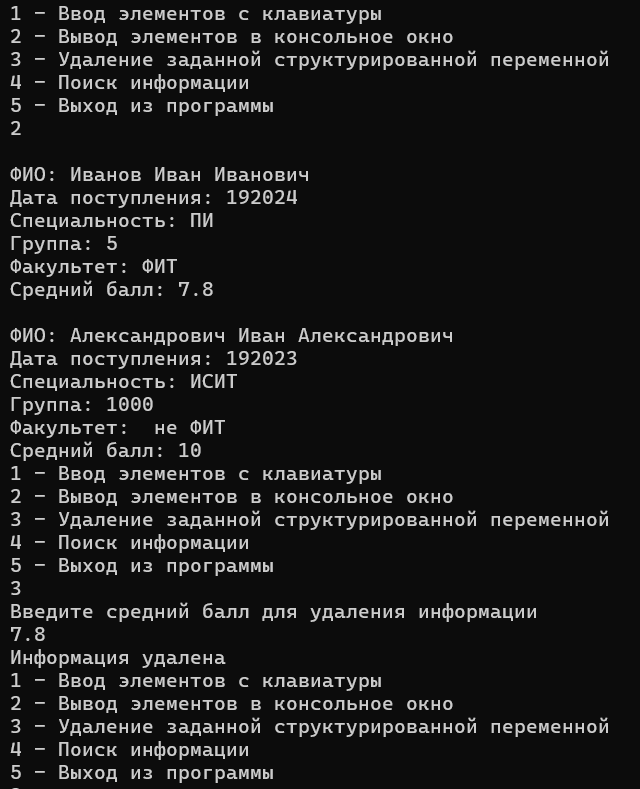
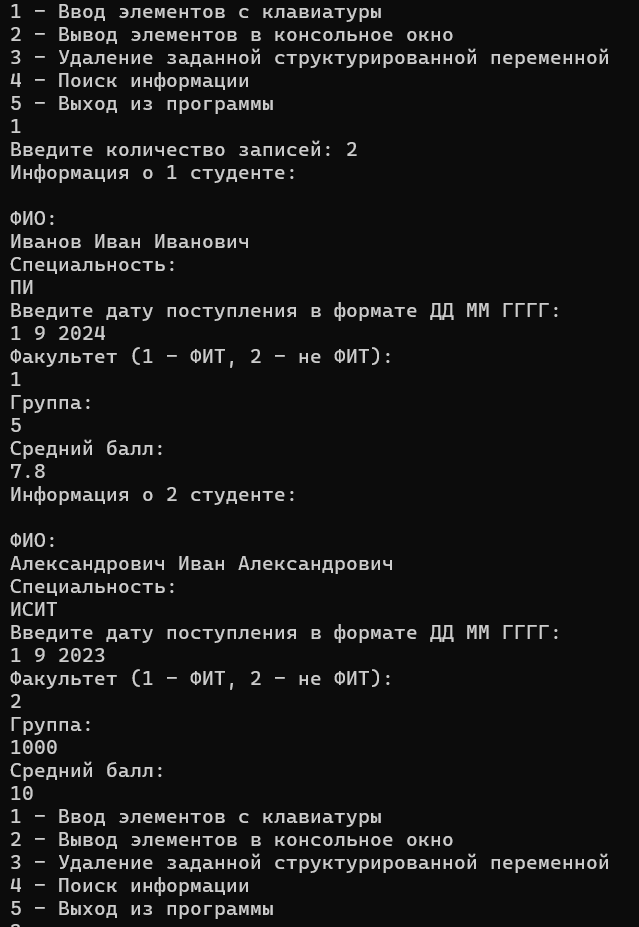
if (f)

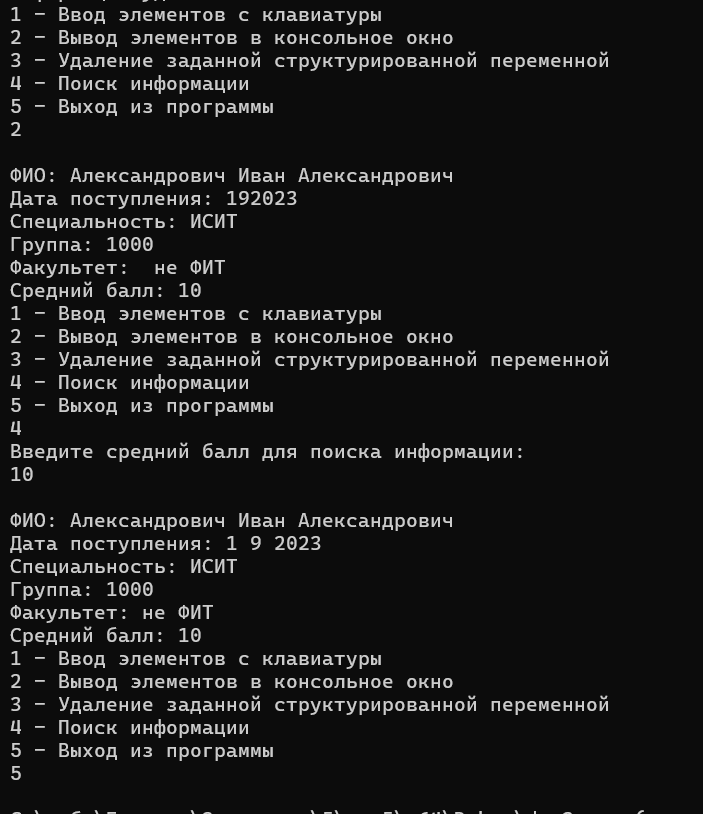
{

cout <<"Информация не найдена!"<< endl;

}

}





**Государство**. Наименование, столица, численность населения, площадь, фамилия президента. Выбор государства по названию.

#include<iostream>

#include<fstream>

#include<string>

#include<windows.h>

#definesize 10 //подключение макроса для длины массива

usingnamespace std;

void input(); //прототип функции ввода

void output(); // прототип функции вывода

void inFile(); // прототип функции для записи в файл

void outFile(); // прототип функции для чтения из файла

void find(); // прототип функции поиска

unionbuf// объединение

{

char name[50];

char capital[50];

int people;

int square;

char prezident[50];

};

structcountry// структура

{

buf nam;

buf cap;

buf peop;

buf sq;

buf pr;

};

country countries[size];

int choice;

int current\_size;

int main()

{

setlocale(LC\_ALL, "Rus");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

//меню выбора

do

{

cout <<"1 - Ввод элементов с клавиатуры"<< endl;

cout <<"2 - Вывод элементов в консольное окно"<< endl;

cout <<"3 - Запись информации в файл"<< endl;

cout <<"4 - Чтение информации из файла"<< endl;

cout <<"5 - Поиск информации"<< endl;

cout <<"6 - Выход из программы"<< endl;

cin >> choice;

// анализируем choice

switch (choice)

{

case 1:

input();

break;

case 2:

output();

break;

case 3:

inFile();

break;

case 4:

outFile();

break;

case 5:

find();

break;

}

} while (choice != 6);

return 0;

}

void input() // функция ввода

{

int numb; // для хранения количества

int choice; // выбор из перечесления

cout <<"Введите количество записей: "; cin >> numb;

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

{

if (current\_size <size) // условие для проверки наличия места

{

// заполнение информации

cin.ignore();

cout <<"Информация о "<< i + 1 <<" стране: "<< endl;

cout << endl <<"Название: "<< endl;

gets\_s(countries[current\_size].nam.name, 50);

cout <<"Столица: "<< endl;

gets\_s(countries[current\_size].cap.capital, 50);

cout <<"Фамилия президента: "<< endl;

gets\_s(countries[current\_size].pr.prezident, 50);

cout <<"Численность: "<< endl;

cin >> countries[current\_size].peop.people;

cout <<"Площадь: "<< endl;

cin >> countries[current\_size].sq.square;

current\_size++;

}

}

}

void output() // функция вывода

{

// вывод информации

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

{

cout <<"Информация о "<< i + 1 <<" стране: "<< endl;

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

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

cout <<"Численность: "<< countries[i].peop.people << endl;

cout <<"Площадь: "<< countries[i].sq.square << endl;

cout <<"Фамилия президента: "<< countries[i].pr.prezident << endl;

}

}

void inFile() // функция записи в файл

{

ofstream file1("FILE1.txt");

if (file1)

{

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

{

file1 <<"Информация о "<< i + 1 <<" стране: "<< endl;

file1 << endl <<"Название: "<< countries[i].nam.name << endl;

file1 <<"Столица: "<< countries[i].cap.capital << endl;

file1 <<"Численность: "<< countries[i].peop.people << endl;

file1 <<"Площадь: "<< countries[i].sq.square << endl;

file1 <<"Фамилия президента: "<< countries[i].pr.prezident << endl;

}

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

file1.close();

}

else

cout <<"Не удалось открыть файл!"<< endl;

}

void outFile() // функция чтения из файла

{

ifstream file1("FILE1.txt");

string line;

while (getline(file1, line))

{

cout << line << endl;

}

file1.close();

}

void find() // функция поиска

{

string line;

bool f = true;

cout <<"Введите фамилию президента: ";

cin.ignore();

getline(cin, line);

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

{

if (line == countries[i].pr.prezident)

{

cout <<"Информация о "<< i + 1 <<" стране: "<< endl;

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

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

cout <<"Численность: "<< countries[i].peop.people << endl;

cout <<"Площадь: "<< countries[i].sq.square << endl;

cout <<"Фамилия президента: "<< countries[i].pr.prezident << endl;

f = false;

}

}

if (f)

{

cout <<"Информация не найдена!"<< endl;

}

}

