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:

Рауба Арсений

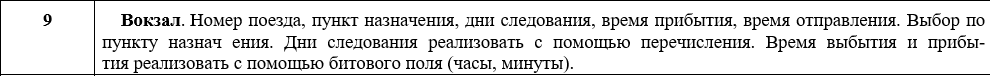
Студент 1 курса 8 группы

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 №9



#include <iostream>

#include <Windows.h>

using namespace std;

void railway\_station();

void print();

void search();

void delete\_info();

enum Edays{ //перечисление

monday=1, tuesday,wednesday,thusday,friday,saturday,sunday

}day;

struct Time{ //битовое поле для времени

int hours: 4;

int minutes: 4;

};

struct Train { //структура для хранения информации

string number;

string destination;

string days;

string departure\_time;

string arrival\_time;

};

const int maxTrains = 50;

int trainCount = 1;

Train trains[maxTrains];

void railway\_station() {//ф-ция для ввода

setlocale(LC\_ALL, "rus");

cout << "Ввод информации: ";

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

cout << "Введите номер поезда:";

cin >> trains[i].number;

cout << "Введите место прибывания:";

cin >> trains[i].destination;

int choice;

cout << "Введите день (1 - пн, 2 - вт, 3 - ср, 4 - чт, 5 - пт, 6 - сб, 7 - вс):";

cin >> choice;

switch(choice) {

case 1:

trains[i].days = monday;

break;

case 2:

trains[i].days = tuesday;

break;

case 3:

trains[i].days = wednesday;

break;

case 4:

trains[i].days = thusday;

break;

case 5:

trains[i].days = friday;

break;

case 6:

trains[i].days = saturday;

break;

case 7:

trains[i].days = sunday;

break;

default:

cout << "Error" << endl;

break;

}

cout << "Введите время отправления:";

cin >> trains[i].departure\_time;

cout << "Введите прибывания:";

cin >> trains[i].arrival\_time;

}

}

void print() {//ф-ция для вывода

setlocale(LC\_ALL, "rus");

int sw, n;

cout << "1-вывод 1 расписания" << endl;

cout << "2-вывод всех расписаний" << endl;

cin >> sw;

if (sw == 1) {

cout << "Номер поезда: ";

cout << trains[trainCount].number << endl;

cout << "Место назначения: ";

cout << trains[trainCount].destination << endl;

cout << "День: ";

cout << trains[trainCount].days << endl;

cout << "Время отправления: ";

cout << trains[trainCount].departure\_time << endl;

cout << "Время прибытия: ";

cout << trains[trainCount].arrival\_time << endl;

}

if (sw == 2) {

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

cout << "Расписание номер: " << i + 1;

cout << "Номер поезда: ";

cout << trains[i].number << endl;

cout << "Место назначения: ";

cout << trains[i].destination << endl;

cout << "День: ";

cout << trains[i].days << endl;

cout << "Время отправления: ";

cout << trains[i].departure\_time << endl;

cout << "Время прибытия: ";

cout << trains[i].arrival\_time << endl;

}

}

}

void search() {//ф-ция для поиска

setlocale(LC\_ALL, "rus");

string nameStation;

bool found = false;

if (trainCount == 0)

{

cout << "Вы еще не ввели ни одного студента" << endl << endl;

return;

}

cout << "Введите фамилию студента, о котором вам нужна информация: ";

cin >> nameStation;

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

{

if (trains[i].destination == nameStation) {

found = true;

cout << "Номер поезда: ";

cout << trains[trainCount].number << endl;

cout << "Место назначения: ";

cout << trains[trainCount].destination << endl;

cout << "День: ";

cout << trains[trainCount].days << endl;

cout << "Время отправления: ";

cout << trains[trainCount].departure\_time << endl;

cout << "Время прибытия: ";

cout << trains[trainCount].arrival\_time << endl;

break;

}

}

if (!found)

{

cout << "Поезд не найден ";

}

}

void delete\_info() {//ф-ция для удаления

setlocale(LC\_ALL, "rus");

if (trainCount == 0)

{

cout << "Вы еще не ввели ни одного поезда" << endl << endl;

return;

}

int delchoice;

cout << "1 - удалить информацию об определенном поезде" << endl

<< "2 - удалить информацию о всех записанных поездах" << endl

<< "Выберите вариант выполнения программы: ";

cin >> delchoice;

if (delchoice == 1)

{

int del;

cout << "Какой поезд хотите удалить";

cin >> del;

for (int i = (del - 1); i < trainCount; i++)

{

trains[i] = trains[i + 1];

}

trainCount--;

cout << "Информация о выбранном поезде успешно удалена!" << endl << endl;

}

if (delchoice == 2)

{

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

{

trains[i].number[0] = '\0';

trains[i].destination[0] = '\0';

trains[i].days[0] = '\0';

trains[i].departure\_time[0] = '\0';

trains[i].arrival\_time[0] = '\0';

}

trainCount = 0;

cout << "Информация обо всех поездах успешно удалена! " << endl << endl;

}

}

int main() {//основная функция

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

setlocale(LC\_ALL, "rus");

int choice;

do {//вывод меню выбора

cout << "1 - добавить расписание поезда" << endl;

cout << "2 - вывод введенных расписаний" << endl;

cout << "3 - удалить введенные расписания" << endl;

cout << "4 - поиск по пункту назначения" << endl;

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

cout << "Введите вариант выполнения программы: ";

cin >> choice;

cout << endl;

switch (choice) {//обработка выбора

case 1:

railway\_station();

break;

case 2:

print();

break;

case 3:

delete\_info();

break;

case 4:

search();

break;

case 5:

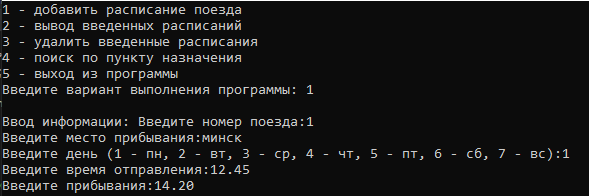
system("exit");

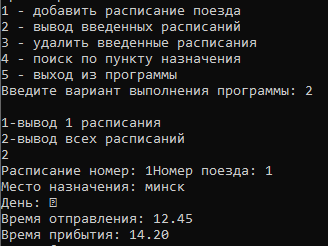
break;

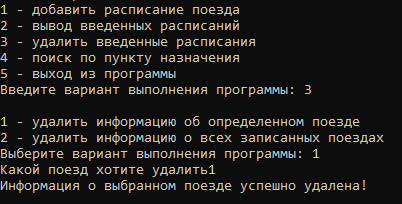
}

} while (choice != 7);

}







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 №9



#include <iostream>

#include <fstream>

#include <Windows.h>

using namespace std;

#define size 30

//Обьявление функций

void enter();//для ввода

void print();//для вывода

void clear();//для очистки

void search();//для поиска

void write();//для записи в файл

void read();//для чтения из файла

union student { // объединение

char name[10];

char secName[10];

char surname[10];

char subjects[100];

int score[3];

float average\_score;

};

//структура для хранения информации о студенте

struct Student {

student name;

student secName;

student surname;

student subjects;

student score;

student average\_score;

};

struct Student list\_of\_student[size];

struct Student del;

int current\_size = 0, choice;

void enter() { //функция для ввода

setlocale(LC\_ALL, "rus");

cout << "Ввод информации: ";

if (current\_size < size) {

cout << "Строка номер: ";

cout << current\_size + 1 << endl;

cout << "Введите фамилию:";

cin >> list\_of\_student[current\_size].secName.secName;

cout << "Введите имя:";

cin >> list\_of\_student[current\_size].name.name;

cout << "Введите отчество:";

cin >> list\_of\_student[current\_size].surname.surname;

cout << "Введите предметы:";

cin >> list\_of\_student[current\_size].subjects.subjects;

cout << "Введите отметки:";

for (int i = 0; i < 3; i++) {

cin >> list\_of\_student[current\_size].score.score[i];

}

current\_size++;

}

else cout << "Введено максимальное кол-во строк";

}

void print() {//функция для вывода

setlocale(LC\_ALL, "rus");

int sw, n;

cout << "1-вывод 1 студента" << endl;

cout << "2-вывод всех студентов" << endl;

cin >> sw;

if (sw == 1) {

cout << "Номер выводимого студента " << endl; cin >> n; cout << endl;

cout << "ФИО: ";

cout << list\_of\_student[n - 1].secName.secName << " " << list\_of\_student[n - 1].name.name << " " << list\_of\_student[n - 1].surname.surname << endl;

cout << "Предметы: ";

cout << list\_of\_student[n - 1].subjects.subjects << endl;

cout << "Отметки: ";

for (int i = 0; i < 3; i++) {

cout << list\_of\_student[n - 1].score.score[i];

}

cout << endl;

for (int i=0; i < 3; i++) {

list\_of\_student[n - 1].average\_score.average\_score = list\_of\_student[n - 1].average\_score.average\_score + list\_of\_student[n - 1].score.score[i];

}

cout << "Средний балл: " << (list\_of\_student[n - 1].average\_score.average\_score)/3 << endl;

}

if (sw == 2) {

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

{

cout << "ФИО: ";

cout << list\_of\_student[i].secName.secName << " " << list\_of\_student[i].name.name << " " << list\_of\_student[i].surname.surname << endl;

cout << "Предметы: ";

cout << list\_of\_student[i].subjects.subjects << endl;

cout << "Отметки: ";

for (int i = 0; i < 3; i++) {

cout << list\_of\_student[i].score.score[i];

}

cout << endl;

for (int i=0; i < 3; i++) {

list\_of\_student[i].average\_score.average\_score = list\_of\_student[i].average\_score.average\_score + list\_of\_student[i].score.score[i];

}

cout << "Средний балл: " << list\_of\_student[n - 1].average\_score.average\_score << endl;

}

}

}

void clear() {//функция для очистки

setlocale(LC\_ALL, "rus");

if (current\_size == 0)

{

cout << "Вы еще не ввели ни одного ученика" << endl << endl;

return;

}

int delchoice;

cout << "1 - удалить информацию об определенной ученике" << endl

<< "2 - удалить информацию о всех записанных учениках" << endl

<< "Выберите вариант выполнения программы: ";

cin >> delchoice;

if (delchoice == 1)

{

int del;

cout << "Какого студента хотите удалить";

cin >> del;

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

{

list\_of\_student[i] = list\_of\_student[i + 1];

}

current\_size--;

cout << "Информация о выбранном студенте успешно удалена!" << endl << endl;

}

if (delchoice == 2)

{

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

{

list\_of\_student[i].secName.secName[0] = '\0';

list\_of\_student[i].name.name[0] = '\0';

list\_of\_student[i].surname.surname[0] = '\0';

list\_of\_student[i].subjects.subjects[0] = '\0';

list\_of\_student[i].average\_score.average\_score = 0;

}

current\_size = 0;

cout << "Информация обо всех учениках успешно удалена! " << endl << endl;

}

}

void search() {//функция для поиска

setlocale(LC\_ALL, "rus");

char stud[30];

bool found = false;

if (current\_size == 0)

{

cout << "Вы еще не ввели ни одного ученика" << endl << endl;

return;

}

cout << "Введите фамилию ученика, о котором вам нужна информация: ";

cin >> stud;

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

{

if (strcmp(list\_of\_student[i].secName.secName, stud) == 0)

{

found = true;

cout << "ФИО: ";

cout << list\_of\_student[i].secName.secName << " " << list\_of\_student[i].name.name << " " << list\_of\_student[i].surname.surname << endl;

cout << "Предметы: ";

cout << list\_of\_student[i].subjects.subjects << endl;

cout << "Отметки: ";

for (int i = 0; i < 3; i++) {

cout << list\_of\_student[i].score.score[i];

}

cout << endl;

cout << "Средний балл: ";

cout << list\_of\_student[i].average\_score.average\_score << endl;

break;

}

}

if (!found)

{

cout << "Ученик с фамилией " << stud << " не найден" << endl << endl;

}

}

void write() {//функция для записи в файл

setlocale(LC\_ALL, "rus");

if (current\_size == 0)

{

cout << "Вы еще не ввели ни одного ученика" << endl << endl;

return;

}

ofstream fout("file.txt");

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

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

return;

}

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

fout << "ФИО: ";

fout << list\_of\_student[i].secName.secName << list\_of\_student[i].name.name << list\_of\_student[i].surname.surname << endl;

fout << "ФИО: ";

fout << list\_of\_student[i].secName.secName << " " << list\_of\_student[i].name.name << " " << list\_of\_student[i].surname.surname << endl;

fout << "Предметы: ";

fout << list\_of\_student[i].subjects.subjects << endl;

fout << "Отметки: ";

for (int i = 0; i < 3; i++) {

fout << list\_of\_student[i].score.score[i];

}

fout << endl;

for (int i=0; i < 3; i++) {

list\_of\_student[i].average\_score.average\_score = list\_of\_student[i].average\_score.average\_score + list\_of\_student[i].score.score[i];

}

fout << "Средний балл: " << list\_of\_student[i].average\_score.average\_score << endl;;

}

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

fout.close();

}

void read() {//функция для чтения из файла

setlocale(LC\_ALL, "rus");

char buff[50];

ifstream fin("file.txt");

if (!fin.is\_open())

{

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

return;

}

while (fin.getline(buff, 50))

{

cout << buff << endl;

}

fin.close();

}

int main() {//основная функция

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

setlocale(LC\_ALL, "rus");

int choice;

do {//вывод меню выбора

cout << "1 - добавить нового ученика" << endl;

cout << "2 - вывод введенных учеников" << endl;

cout << "3 - удалить введенных учеников" << endl;

cout << "4 - вывод информации по фамилии" << endl;

cout << "5 - запись введенных данных в файл" << endl;

cout << "6 - прочитать данные из файла" << endl;

cout << "7 - выход из программы" << endl;

cout << "Введите вариант выполнения программы: ";

cin >> choice;

cout << endl;

switch (choice) {//обработка выбора

case 1:

enter();

break;

case 2:

print();

break;

case 3:

clear();

break;

case 4:

search();

break;

case 5:

write();

break;

case 6:

read();

break;

case 7:

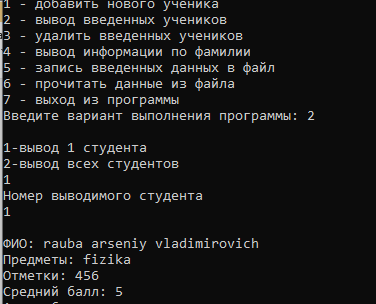
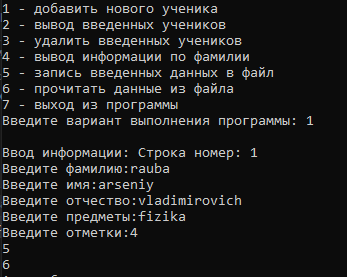
system("exit");

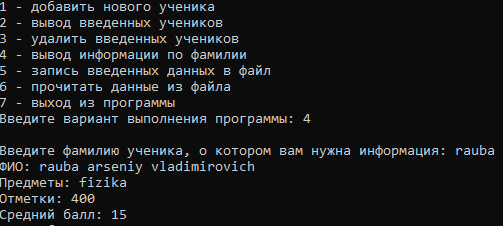
break;

}

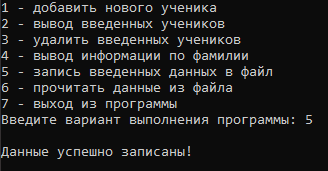
} while (choice != 7);

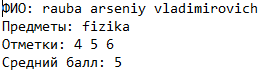
}











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;

}

}

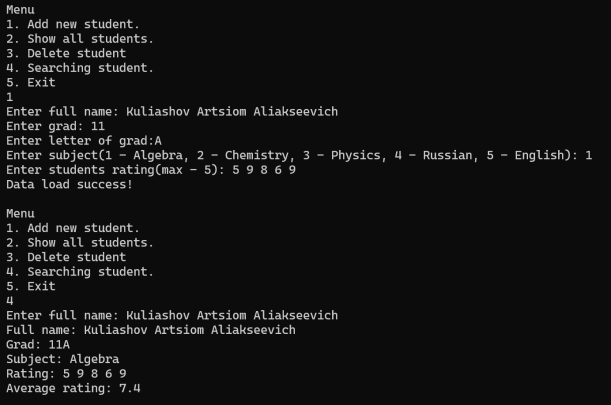
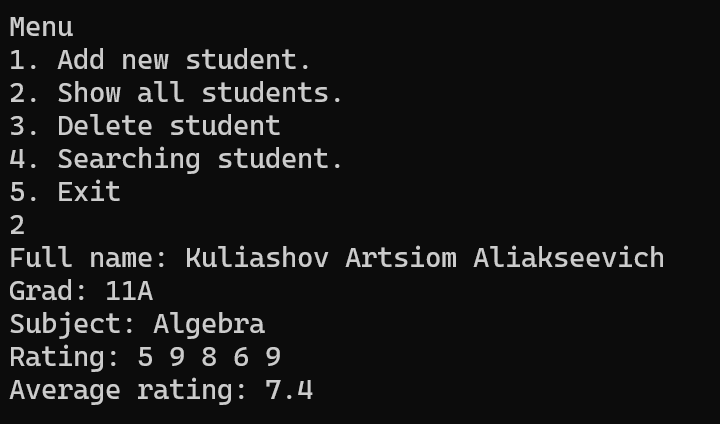
if (!found)

{

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

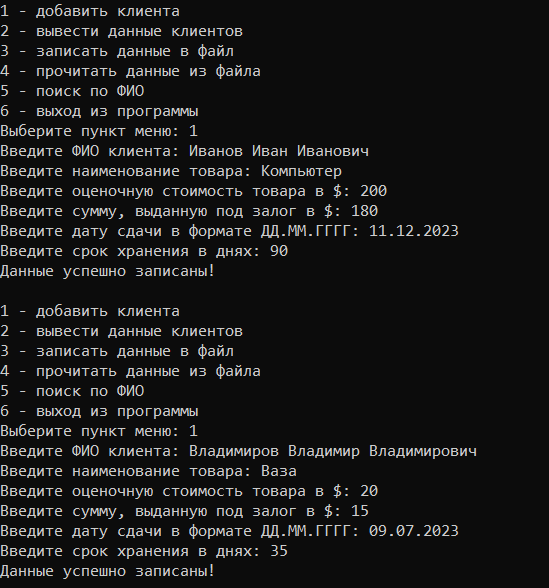
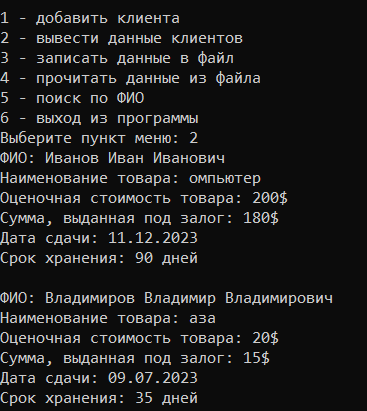
    if (!found)

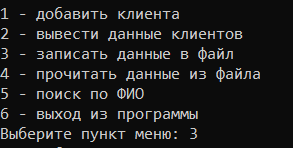
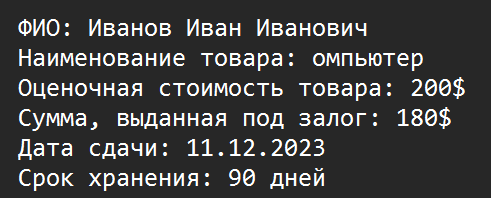
    {

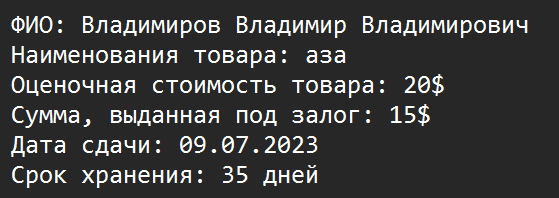
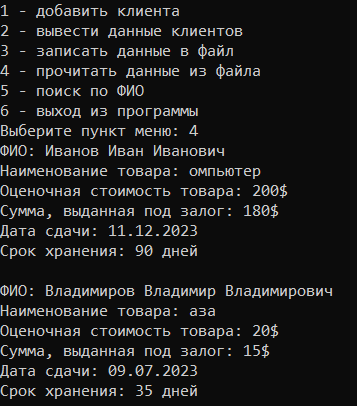
        cout << "Клиент с такими ФИО не найден" << endl << endl;

    }

}

** **

**** 

****

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;

}

}

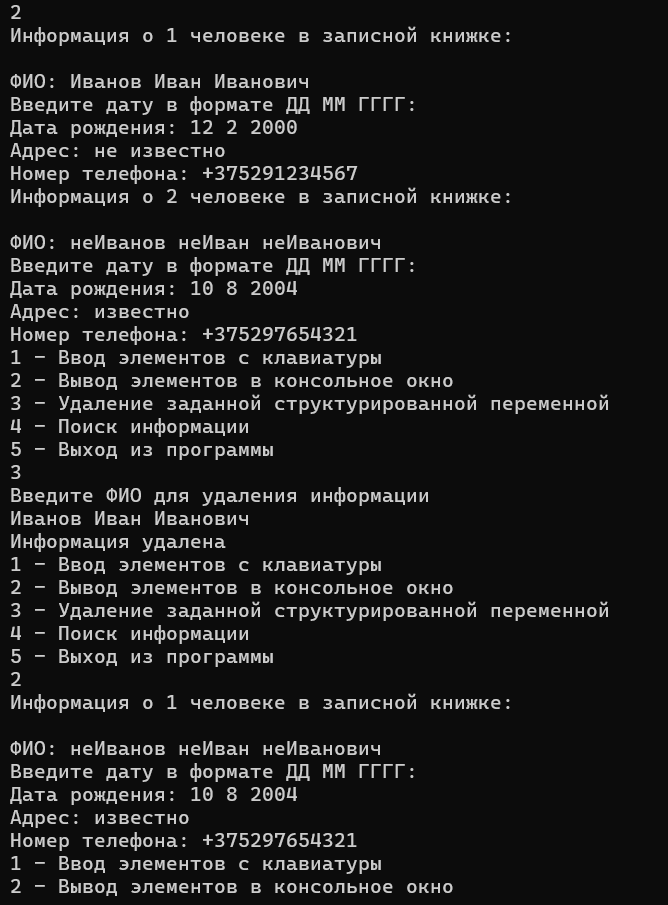
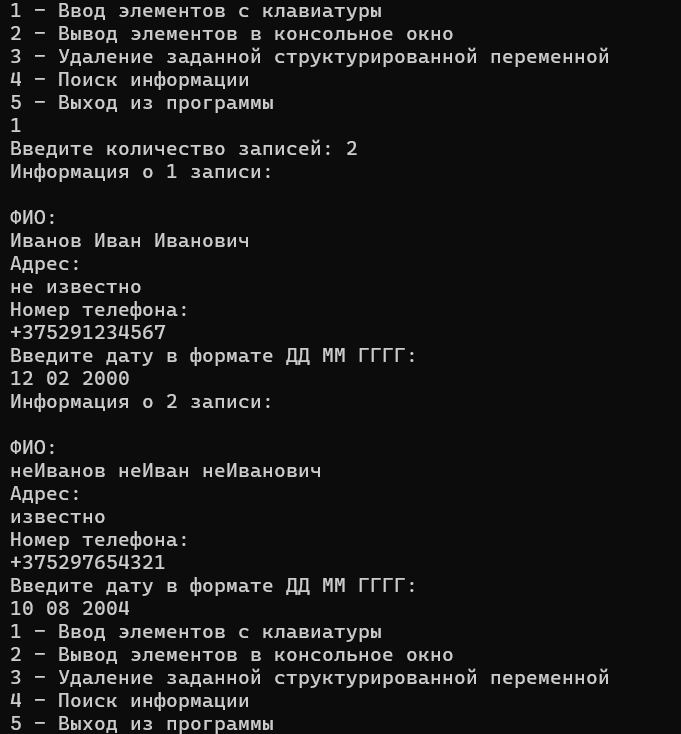
if (f)

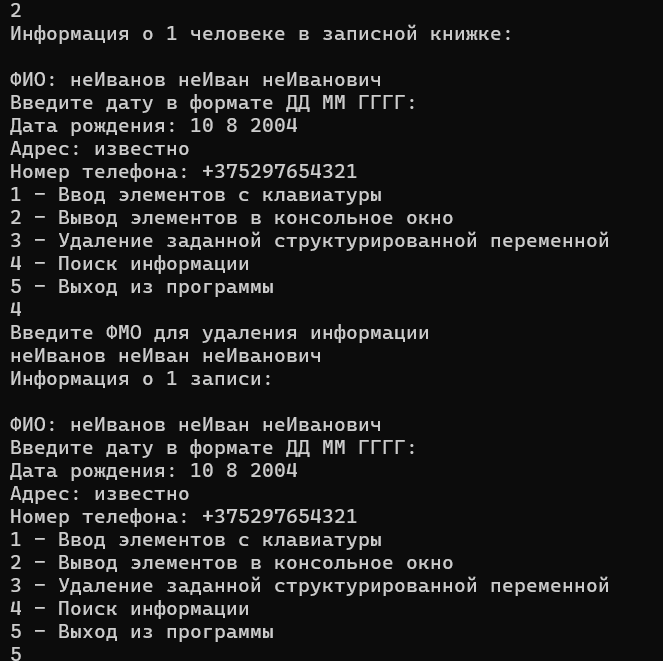
{

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;

}

}

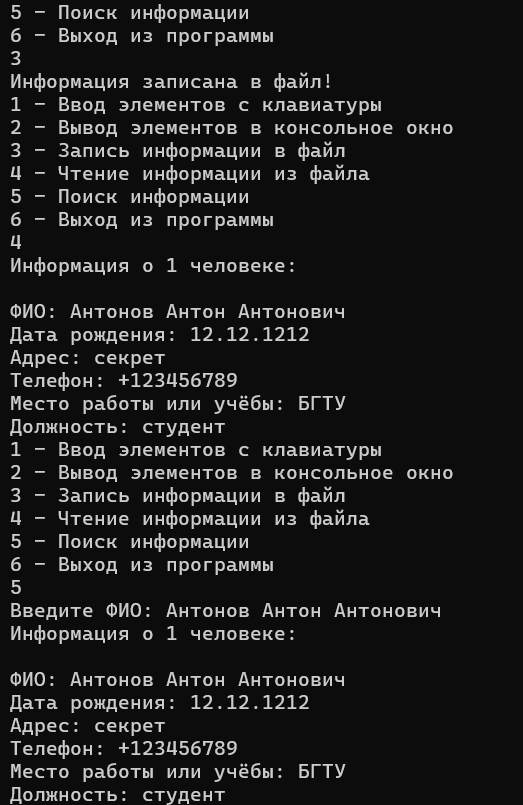
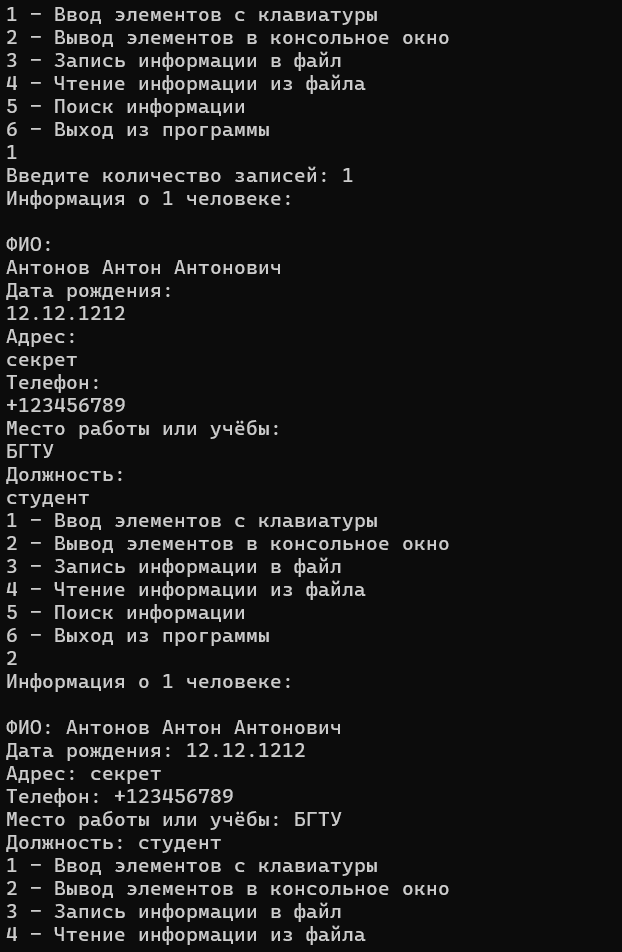
if (f)

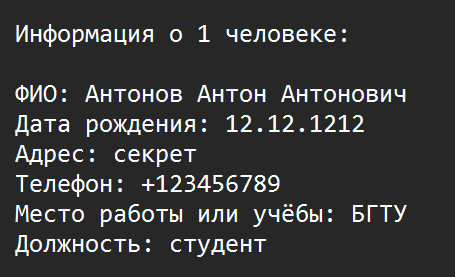
{

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;

}

}

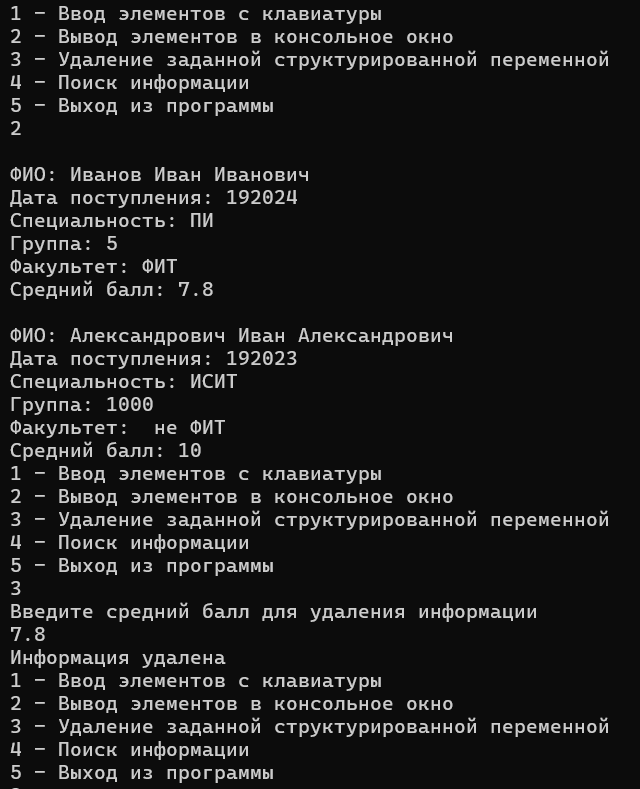
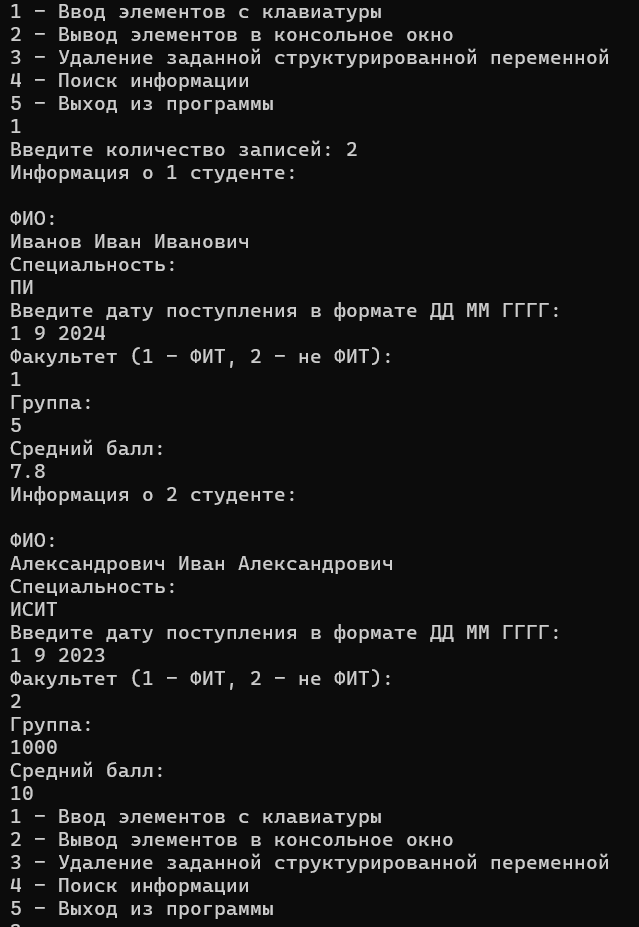
if (f)

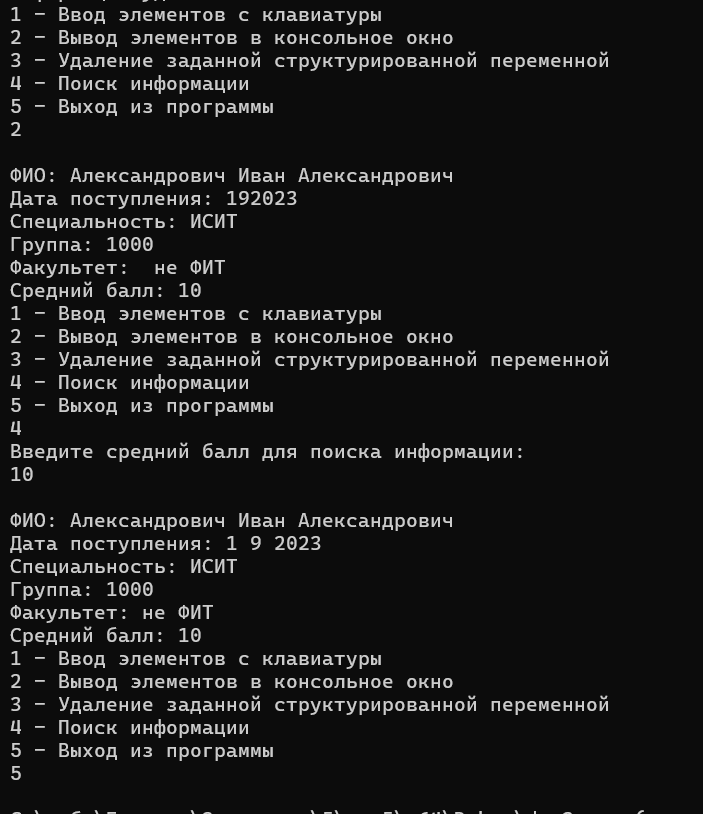
{

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;

}

}

