Министерство образования и науки Кыргызской Республики

Кыргызско-Германский институт прикладной информатики

СРСП №1

по теме «Работа с структурами и массивами в С++»

Выполнила: студентка группы AIN-1-22

Тахирова К.

Проверила: доцент КГИПИ

Картанова А. Д.

Оглавление

[Цель 3](#_Toc178860492)

[Задание 3](#_Toc178860493)

[Описание 3](#_Toc178860494)

[Использование структур и массивов 4](#_Toc178860495)

[Описание функций 5](#_Toc178860496)

[Исходный код программы 9](#_Toc178860497)

[Вывод 12](#_Toc178860498)

# Цель

Целью данной самостоятельной работы является освоение работы со структурами и массивами в языке программирования C++. В рамках выполнения задания предполагается разработать консольное приложение для управления массивом записей, каждая из которых содержит информацию о человеке, включая фамилию, имя, номер телефона и дату рождения.

# Задание

В а р и а н т 1 2

Описать структуру с именем NOTE, содержащую следующие поля:

• фамилия, имя;

• номер телефона;

• дата рождения (массив из трех чисел).

Написать программу, выполняющую следующие действия:

• ввод с клавиатуры данных в массив, состоящий из восьми элементов типа NOTE

(записи должны быть упорядочены по дате рождения);

• вывод на экран информации о человеке, номер телефона которого введен с клавиатуры

(если такого нет, вывести соответствующее сообщение).

Внесенные изменения и обоснования

1. **Расширенный функционал**: В программу были добавлены функции для поиска записей по фамилии и номеру телефона, что повысило её удобство и функциональность.
2. **Проверка ввода**: Реализована логика проверки наличия свободного места в массиве перед добавлением новой записи, что предотвращает переполнение.
3. **Информативные сообщения**: Добавлены сообщения, информирующие пользователя о результатах операций, например, об отсутствии записи при поиске.

# Описание

В рамках данной работы было разработано консольное приложение, которое предоставляет пользователю возможность выполнять несколько ключевых операций с записями о людях. Ниже представлены основные функции и возможности приложения:

1. **Ввод данных о людях**:
   * Пользователь может ввести данные о людях в массив из восьми элементов типа NOTE. Каждая запись включает в себя следующие поля:
     + **Фамилия**: строка, представляющая фамилию человека.
     + **Имя**: строка, представляющая имя человека.
     + **Номер телефона**: строка, содержащая номер телефона.
     + **Дата рождения**: массив из трех целых чисел, представляющий день, месяц и год рождения.
   * Программа запрашивает ввод этих данных поэтапно, что делает процесс интуитивно понятным и удобным.
2. **Упорядочивание записей по дате рождения**:
   * После ввода данных о каждом человеке записи автоматически сортируются по дате рождения. Это позволяет пользователю легко отслеживать и находить людей по их возрасту.
   * Упорядочивание осуществляется на основе сравнения массивов дат, что гарантирует правильное расположение записей в соответствии с хронологией.
3. **Поиск и вывод информации**:
   * Пользователь имеет возможность искать и выводить информацию о человеке по номеру телефона. Для этого достаточно ввести искомый номер в консоль.
   * Если запись с указанным номером телефона найдена, программа выводит полные данные о человеке (фамилию, имя, номер телефона и дату рождения). Если запись отсутствует, программа предоставляет соответствующее сообщение, информируя пользователя о том, что такая запись не найдена.
   * Эта функция улучшает взаимодействие с пользователем, позволяя быстро находить нужную информацию без необходимости просматривать весь массив.

# Использование структур и массивов

Приложение использует структуры для организации данных и массивы для управления записями. Это обеспечивает:

* **Эффективное хранение данных**: Структура NOTE объединяет связанные данные (фамилия, имя, телефон, дата рождения) в единый объект, что упрощает управление и доступ к ним.
* **Гибкость и масштабируемость**: Использование массива позволяет хранить фиксированное количество записей (в данном случае восемь), что может быть легко изменено при необходимости, а также упрощает реализацию функциональности поиска и сортировки.
* **Чистота и читаемость кода**: Применение структур позволяет разделять логику работы с данными и их представление, что делает код более организованным и легким для понимания.

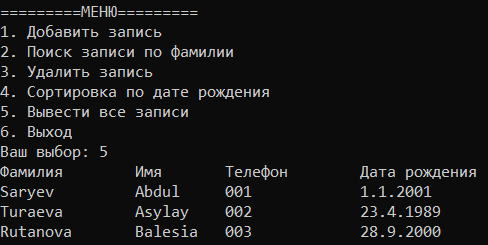
# Описание функций

start(): Основная функция управления, которая предоставляет меню выбора операций и обрабатывает пользовательский ввод.

loadNotes(): Чтение записей из файла.

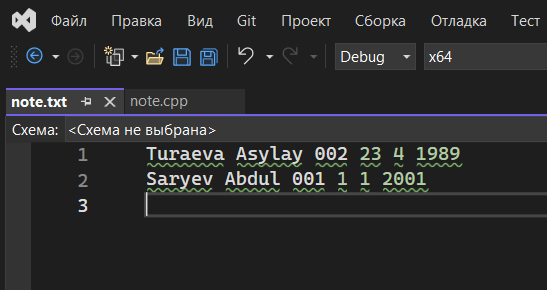
saveNotes(): Сохранение данных в файл.

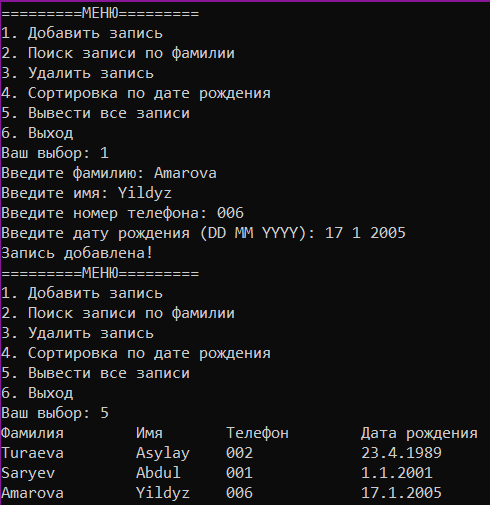
displayNotes(): Форматированный вывод записей



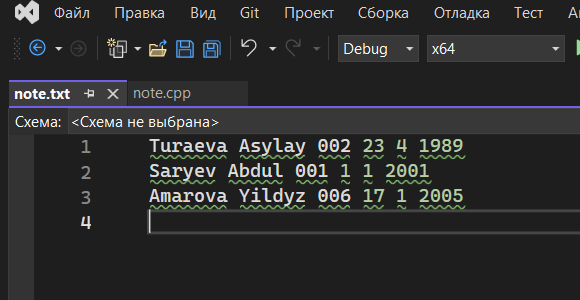
addNote(): Функция для добавления новой записи. Запрашивает ввод данных и добавляет запись в массив, если есть свободное место.

Содержимое файла note.txt до добавления записи:

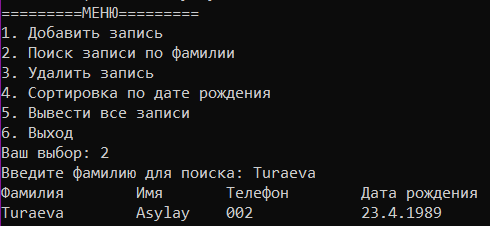




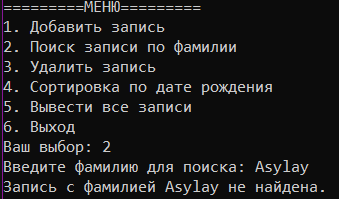
Содержимое файла note.txt после добавления записи:



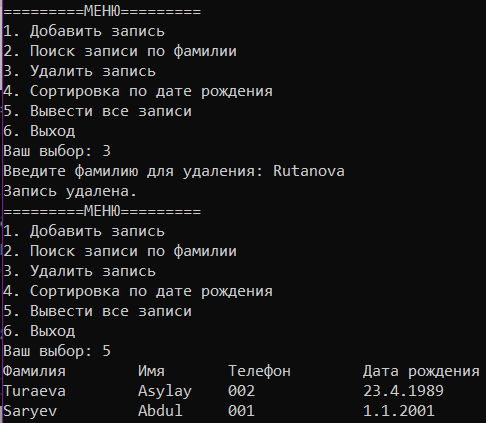
findByLastName(): Функция для поиска записи по фамилии, также возвращает соответствующее сообщение при отсутствии.



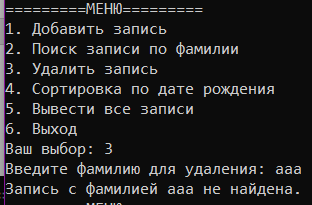
В случае, если контакта с такой фамилией нет:



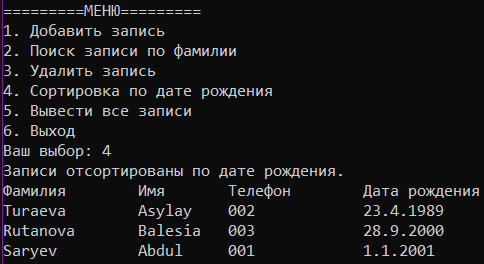
deleteNote(): Удаление записи по фамилии.



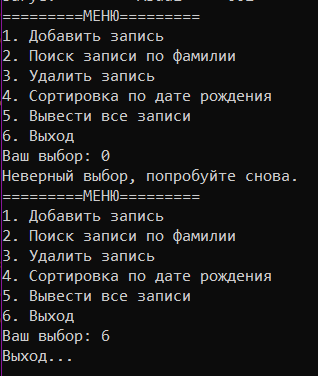
В случае, если контакта с такой фамилией нет:



sortByBirthDate(): Сортировка записей по дате рождения



Выход из программы:



# Исходный код программы

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <sstream>

#include <algorithm>

#include <locale>

#include <iomanip>

using namespace std;

struct Note {

string lastName;

string name;

string phoneNum;

int birthDate[3];

};

// Чтение записей из файла

vector<Note> loadNotes(const string& filename) {

vector<Note> notes;

ifstream file(filename);

string line;

while (getline(file, line)) {

Note note;

stringstream ss(line);

ss >> note.lastName >> note.name >> note.phoneNum

>> note.birthDate[0] >> note.birthDate[1] >> note.birthDate[2];

notes.push\_back(note);

}

file.close();

return notes;

}

// Запись записей в файл

void saveNotes(const string& filename, const vector<Note>& notes) {

ofstream file(filename, ios::app); // Открываем в режиме добавления

for (const auto& note : notes) {

file << note.lastName << " " << note.name << " " << note.phoneNum << " "

<< note.birthDate[0] << " " << note.birthDate[1] << " " << note.birthDate[2] << endl;

}

file.close();

}

// Добавление записи

void addNote(vector<Note>& notes, const string& filename) {

if (notes.size() >= 8) {

cout << "Массив записей полон (8 записей)." << endl;

return;

}

Note note;

cout << "Введите фамилию: ";

cin >> note.lastName;

cout << "Введите имя: ";

cin >> note.name;

cout << "Введите номер телефона: ";

cin >> note.phoneNum;

cout << "Введите дату рождения (DD MM YYYY): ";

cin >> note.birthDate[0] >> note.birthDate[1] >> note.birthDate[2];

notes.push\_back(note);

// Запись новой записи в файл

saveNotes(filename, { note });

cout << "Запись добавлена!" << endl;

}

// Поиск записи по фамилии

void findByLastName(const vector<Note>& notes, const string& lastName) {

bool found = false;

for (const auto& note : notes) {

if (note.lastName == lastName) {

cout << setw(15) << left << "Фамилия"

<< setw(10) << left << "Имя"

<< setw(15) << left << "Телефон"

<< "Дата рождения" << endl;

cout << setw(15) << left << note.lastName

<< setw(10) << left << note.name

<< setw(15) << left << note.phoneNum

<< note.birthDate[0] << "." << note.birthDate[1] << "." << note.birthDate[2] << endl;

found = true;

}

}

if (!found) {

cout << "Запись с фамилией " << lastName << " не найдена." << endl;

}

}

// Удаление записи

void deleteNote(vector<Note>& notes, const string& lastName) {

auto it = remove\_if(notes.begin(), notes.end(), [&](const Note& note) {

return note.lastName == lastName;

});

if (it != notes.end()) {

notes.erase(it, notes.end());

// Перезаписываем файл без удаленной записи

ofstream file("note.txt");

for (const auto& note : notes) {

file << note.lastName << " " << note.name << " " << note.phoneNum << " "

<< note.birthDate[0] << " " << note.birthDate[1] << " " << note.birthDate[2] << endl;

}

file.close();

cout << "Запись удалена." << endl;

}

else {

cout << "Запись с фамилией " << lastName << " не найдена." << endl;

}

}

// Сортировка по дате рождения

void sortByBirthDate(vector<Note>& notes) {

sort(notes.begin(), notes.end(), [](const Note& a, const Note& b) {

if (a.birthDate[2] != b.birthDate[2]) return a.birthDate[2] < b.birthDate[2];

if (a.birthDate[1] != b.birthDate[1]) return a.birthDate[1] < b.birthDate[1];

return a.birthDate[0] < b.birthDate[0];

});

cout << "Записи отсортированы по дате рождения." << endl;

}

void displayNotes(const vector<Note>& notes) {

cout << setw(15) << left << "Фамилия"

<< setw(10) << left << "Имя"

<< setw(15) << left << "Телефон"

<< "Дата рождения" << endl;

for (const auto& note : notes) {

cout << setw(15) << left << note.lastName

<< setw(10) << left << note.name

<< setw(15) << left << note.phoneNum

<< note.birthDate[0] << "." << note.birthDate[1] << "." << note.birthDate[2] << endl;

}

}

// Основное меню программы

void start(const string& filename) {

vector<Note> notes = loadNotes(filename);

int choice;

bool isRunning = true;

while (isRunning) {

cout << "=========МЕНЮ========="

<< "\n1. Добавить запись"

<< "\n2. Поиск записи по фамилии"

<< "\n3. Удалить запись"

<< "\n4. Сортировка по дате рождения"

<< "\n5. Вывести все записи"

<< "\n6. Выход"

<< "\nВаш выбор: ";

cin >> choice;

switch (choice) {

case 1:

addNote(notes, filename);

break;

case 2: {

string lastName;

cout << "Введите фамилию для поиска: ";

cin >> lastName;

findByLastName(notes, lastName);

break;

}

case 3: {

string lastName;

cout << "Введите фамилию для удаления: ";

cin >> lastName;

deleteNote(notes, lastName);

break;

}

case 4:

sortByBirthDate(notes);

displayNotes(notes); // Показываем отсортированные записи

break;

case 5:

displayNotes(notes);

break;

case 6:

cout << "Выход..." << endl;

isRunning = false;

break;

default:

cout << "Неверный выбор, попробуйте снова." << endl;

}

}

}

int main() {

setlocale(LC\_ALL, "Russian");

const string filename = "note.txt";

start(filename);

return 0;

}

# Вывод

В результате выполнения данной работы мы научились:

* эффективно использовать структуры для организации данных;
* работать с массивами структур и управлять записями;
* реализовывать функции поиска и обработки данных в консольных приложениях.

Работа продемонстрировала значимость структур и массивов при решении задач, связанных с хранением и обработкой данных, а также углубила понимание принципов проектирования программного обеспечения на C++.