МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГАОУ ВО «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ИНФОРМАЦИОННЫХ

ТЕХНОЛОГИЙ И ТЕЛЕКОММУНИКАЦИЙ

КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ

**Лабораторная работа №2**

по дисциплине

«Информационные технологии и программирования»

**Выполнил:**

Иванов Даниил Евгеньевич

Студент 1 курса группы \_ПИН-б-о-22-1

Направления подготовки

09.03.03 Прикладная информатика

очной формы обучения

Ставрополь, 2023 г.

Тема: Классы.

Цель работы: изучить базовые понятия (классы, подклассы и методы) Реализовать фундаментальные принципы объектно-ориентированного программирования.

Выполнение работы:

Вариант 16

Описать класс «студенческая группа». Предусмотреть возможность

работы с переменным числом студентов, поиска студента по какому-либо признаку (например, по фамилии, дате рождения или номеру телефона), добавления и удаления записей, сортировки по разным полям. Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

// Определение класса "Студент"

class Student {

private:

string firstName;

string lastName;

string dateOfBirth;

string phoneNumber;

public:

// Конструктор для инициализации полей студента

Student(string first, string last, string dob, string phone)

: firstName(first), lastName(last), dateOfBirth(dob), phoneNumber(phone) {}

// Метод для вывода информации о студенте

void displayInfo() const {

cout << "Имя Фамилия: " << firstName << " " << lastName << "\n";

cout << "Дата рождения: " << dateOfBirth << "\n";

cout << "Номер телефона: " << phoneNumber << "\n";

}

// Методы для получения атрибутов студента

string getFirstName() const { return firstName; }

string getLastName() const { return lastName; }

string getDateOfBirth() const { return dateOfBirth; }

string getPhoneNumber() const { return phoneNumber; }

};

// Определение класса "Студенческая группа"

class StudentGroup {

private:

vector<Student> students;

public:

// Метод для добавления студента в группу

void addStudent(const Student& student) {

students.push\_back(student);

}

// Метод для удаления студента из группы по фамилии

void removeStudent(const string& lastName) {

students.erase(remove\_if(students.begin(), students.end(),

[lastName](const Student& student) {

return student.getLastName() == lastName;

}),

students.end());

}

// Метод для поиска студента по фамилии

Student\* findStudentByLastName(const string& lastName) {

for (auto& student : students) {

if (student.getLastName() == lastName) {

return &student;

}

}

return nullptr; // Возвращаем nullptr, если студент не найден

}

// Метод для сортировки студентов по фамилии

void sortStudentsByLastName() {

sort(students.begin(), students.end(),

[](const Student& a, const Student& b) {

return a.getLastName() < b.getLastName();

});

}

// Метод для вывода всех студентов в группе

void displayAllStudents() const {

for (const auto& student : students) {

student.displayInfo();

cout << "\n";

}

}

};

int main() {

setlocale(LC\_ALL, "Russian");

StudentGroup group;

int choice;

do {

cout << "Меню:\n";

cout << "1. Добавить студента\n";

cout << "2. Удалить студента\n";

cout << "3. Найти студента по фамилии\n";

cout << "4. Отсортировать студентов по фамилии\n";

cout << "5. Отобразить всех учащихся\n";

cout << "6. Выход\n";

cout << "Введите свой выбор: ";

cin >> choice;

switch (choice) {

case 1: {

string first, last, dob, phone;

cout << "Введите имя: ";

cin >> first;

cout << "Введите фамилию: ";

cin >> last;

cout << "Введите дату рождения: ";

cin >> dob;

cout << "Введите номер телефона: ";

cin >> phone;

group.addStudent(Student(first, last, dob, phone));

break;

}

case 2: {

string lastName;

cout << "Введите фамилию для удаления: ";

cin >> lastName;

group.removeStudent(lastName);

break;

}

case 3: {

string lastName;

cout << "Введите фамилию, чтобы найти: ";

cin >> lastName;

Student\* foundStudent = group.findStudentByLastName(lastName);

if (foundStudent != nullptr) {

cout << "Найден студет:\n";

foundStudent->displayInfo();

}

else {

cout << "Студент не найден.\n";

}

break;

}

case 4:

group.sortStudentsByLastName();

cout << "Студенты отсортированы по фамилии.\n";

break;

case 5:

cout << "Все студенты:\n";

group.displayAllStudents();

break;

case 6:

cout << "Выходящий\n";

break;

default:

cout << "Неверный выбор. Пожалуйста, попробуйте снова.\n";

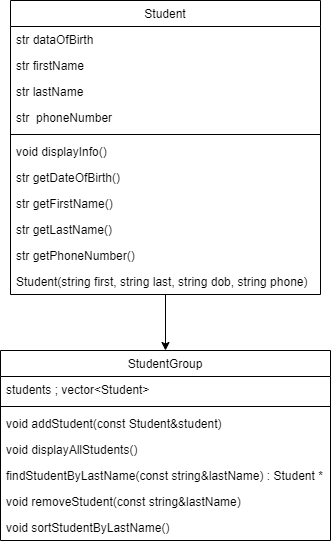
break;

}

} while (choice != 6);

return 0;

}



Ссылка на полностью сделанные задания на github:

Вывод: изучил базовые понятия (классы, подклассы и методы) Реализовал фундаментальные принципы объектно-ориентированного программирования.