# МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования «Кемеровский государственный университет»

институт цифры

кафедра цифровых технологий

# ОТЧЕТ

о выполнении

Лабораторной работы №1 **«Многофайловые проекты на С++»**

по курсу «Языки программирования»

студента 1 курса

Коник Ильи Николаевича

(ФИО полностью)

направление подготовки 02.03.03 Математическое обеспечение и администрирование информационных систем

направленность (профиль) подготовки «Интеллектуальные информационные системы и анализ данных»

направление подготовки 02.03.02 Фундаментальная информатика и информационные технологии

направленность (профиль) подготовки «Информатика и компьютерные науки»

Преподаватель:

Дуванов И. О.

Кемерово 2025

**Тема: Многофайловые проекты на С++** *(3 балла)*

## Задание

1. В соответствии с вариантами задания и приведённой ниже спецификацией реализуйте программу на языке C++.
2. Разработайте тестовые задания, и протестируйте программу одним из методов тестирования.

## Требования к оформлению программ:

* 1. **Содержание**. Программа должна делать только то, что предусмотрено заданием. Не нужно выполнять лишнюю работу.
  2. **Спецификация.** В преамбуле программы в комментариях указывать сведения:
     + Кто выполнил.
     + Что делает программа (кратко).
     + Что на входе, имена входных файлов указываются.
     + Что на выходе (что является результатов работы программы).

# Ввод и вывод

* + - Приглашение пользователю. (Например, сколько чисел, какого типа и через какой разделитель нужно вводить).
    - Контрольный вывод (все введенные данные выводить на экран, и только после этого выполнять необходимые вычисления.)
    - «Защита от дурака». Проверять вводимые данные на корректность. Например, если необходимо считать количество чего – то, то эта величина не может быть отрицательной и т.д.
  1. **Структура кода.** Набираемый код должен быть хорошо структурированным. Использовать:
     + Отступы.
     + Комментарии – поясняют решение программы.
     + Осмысленные названия переменных.
     + Пояснения о назначении переменных в комментариях (кроме счетчиков).

# Декомпозиция кода

* + - Функциональная
    - Объектно-ориентированная

# Многофайловые проекты

* + - Классы определять в отдельном h-файле, а все методы классов – в

соответствующем cpp - файле. Созданный заголовочный файл подключать к проекту.

* + - Методы класса реализовывать не в определении класса.
    - cpp-файл и соответствующий ему h-файл называть одинаково. В качестве названия выбирать имя того класса, который определен в соответствующем модуле.

## Работа в классе:

* В файле lab1.cpp приведен пример работы со структурой Time. Проанализируйте работу программы. Измените код, сделав функции Normalize, AddSeconds, Print методами структуры Time.
* Выполните композицию кода file.cpp на три файла: Time.h (с определением структуры), Time.cpp (с реализацией методов структуры), main.cpp (файл с главной программой, в которой выполняется работа с одним объектом структурного типа Time).
* Проанализируйте работу измененной программы.

## Задание:

* Опишите свой объект с помощью структур.
* Предусмотрите набор полей не менее чем из 4 наименований разных типов, хотя бы одно из полей должно быть строкового типа.
* Подготовьте файл с данными об объектах, количество записей в файле считайте неизвестным (для тестирования предусмотрите не менее 5). Создайте массив объектов типа Вашей структуры, инициализируйте его данными из файла.
* Реализуйте один из простых методов сортировки (метод пузырька, метод вставки, метод выбора, метод Шелла, шейкерная сортировка, метод Хоара) и отсортируйте массив структур дважды: по одному числовому полю по убыванию и по одному строковому полю по возрастанию.
* Выполните файловую композицию кода.

***Варианты объектов:***

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | Абитуриент | 18. | Заяц |
| 2. | Авиабилет | 19. | Игрушка |
| 3. | Автомобиль | 20. | Кинофильм |
| 4. | Библиотека | 21. | Компьютер |
| 5. | Билет | 22. | Конфета |
| 6. | Велосипед | 23. | Кулинарный рецепт |
| 7. | Вирус | 24. | Лампа |
| 8. | Врач | 25. | Магнитофон |
| 9. | Гостиница | 26. | Кролик |
| 10. | Гриб | 27. | Мяч |
| 11. | Деньги | 28. | Обувь |
| 12. | Диван | 29. | Одежда |
| 13. | Диски | 30. | Озеро |
| 14. | Дом | 31. | Парикмахерская |
| 15. | Дошкольник | 32. | Партия |

|  |  |  |  |
| --- | --- | --- | --- |
| 16. | Елка | 33. | Пассажирский транспорт |
| 17. | Записная книжка | 34. | Печатное издание |

# Критерии оценки:

Если программа компилируется и оформлена по требованиям, то за нее начисляются баллы. Максимально можно получить 3 балла.

правильно описан тип данных – 0,5 баллов.

правильно реализован метод сортировки чисел

* метод пузырька, метод вставки, метод выбора – 0,5 балла
* метод Шелла, шейкерная сортировка, метод Хоара – 1 балл

правильно реализована работа с файлами данных – 0,5 баллов

правильно реализована сортировка по числовому полю – 0,5 баллов правильно реализована сортировка по строковому полю – 0,5 баллов  
  
Работа на паре:  
  
main.cpp

#include <iostream>

#include "Time.h"

using std::cout;

using std::endl;

int main()

{

    Time t;

    t.seconds = 12100;

    t.minutes=0;

    t.hours=0;

    Print(t);

    Normalize(t);

    Print(t);

    AddSeconds(t,60);

    Print(t);

    return 0;

}

Time.cpp

#include "Time.h"

#include <iostream>

using namespace std;

void Normalize(Time& t){

    t.minutes += t.seconds / 60;

    t.seconds %= 60;

    if (t.seconds < 0)

    {

        t.minutes -= 1;

        t.seconds += 60;

    }

    t.hours += t.minutes / 60;

    t.minutes %= 60;

    if (t.minutes < 0)

    {

        t.hours -= 1;

        t.minutes += 60;

    }

    t.hours %= 24;

    if (t.hours < 0)

    {

        t.hours += 24;

    }

}

void AddSeconds(Time t,int s){

    t.seconds += s;

    Normalize(t);

}

void Print(Time t){

    cout<<t.hours<<" : "<<t.minutes<<" : "<<t.seconds<<endl;

}

Time.h

#ifndef TIME\_H

#define TIME\_H

struct Time{

    int hours = 0;

    int minutes = 0;

    int seconds = 0;

};

void Normalize(Time& t);

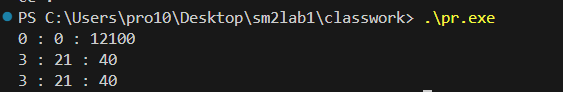
void AddSeconds(Time t,int s);

void Print(Time t);

#endif //TIME\_H







Домашняя работа:  
  
Candy.cpp

#include "Candy.h"

#include <iostream>

#include <fstream>

#include <algorithm>

// Реализация метода для сортировки по калориям (по убыванию)

void Candy::bubbleSortByCalories(std::vector<Candy>& candies) {

    int n = candies.size();

    for (int i = 0; i < n - 1; i++) {

        for (int j = 0; j < n - i - 1; j++) {

            if (candies[j].calories < candies[j + 1].calories) {

                std::swap(candies[j], candies[j + 1]);

            }

        }

    }

}

// Реализация метода для сортировки по названию (по возрастанию)

void Candy::bubbleSortByName(std::vector<Candy>& candies) {

    int n = candies.size();

    for (int i = 0; i < n - 1; i++) {

        for (int j = 0; j < n - i - 1; j++) {

            if (candies[j].name > candies[j + 1].name) {

                std::swap(candies[j], candies[j + 1]);

            }

        }

    }

}

// Реализация метода для вывода информации о конфете

void Candy::print() const {

    std::cout << name << " " << weight << " " << calories << " " << flavor << "\n";

}

// Реализация функции для чтения данных из файла

std::vector<Candy> readCandiesFromFile(const std::string& filename) {

    std::ifstream file(filename);

    std::vector<Candy> candies;

    Candy temp;

    if (!file.is\_open()) {

        std::cerr << "Error: Could not open file " << filename << "\n";

        std::cerr << "Please make sure the file exists and is in the correct directory.\n";

        return candies;

    }

    while (file >> temp.name >> temp.weight >> temp.calories >> temp.flavor) {

        candies.push\_back(temp);

    }

    return candies;

}

Candy.h

#ifndef CANDY\_H

#define CANDY\_H

#include <string>

#include <vector>

// Структура Candy

struct Candy {

    std::string name;       // Название конфеты

    double weight;          // Вес конфеты в граммах

    int calories;           // Количество калорий

    std::string flavor;     // Вкус конфеты

    // Методы для сортировки

    static void bubbleSortByCalories(std::vector<Candy>& candies);

    static void bubbleSortByName(std::vector<Candy>& candies);

    // Метод для вывода информации о конфете

    void print() const;

};

// Функция для чтения данных из файла

std::vector<Candy> readCandiesFromFile(const std::string& filename);

#endif

main.cpp

#include "Candy.h"

#include <iostream>

int main() {

    // Чтение данных из файла

    std::vector<Candy> candies = readCandiesFromFile("candies.txt");

    // Сортировка по калориям (по убыванию)

    Candy::bubbleSortByCalories(candies);

    std::cout << "Sorted by calories (descending):\n";

    for (const auto& candy : candies) {

        candy.print();

    }

    // Сортировка по названию (по возрастанию)

    Candy::bubbleSortByName(candies);

    std::cout << "\nSorted by name (ascending):\n";

    for (const auto& candy : candies) {

        candy.print();

    }

    return 0;

}

candies.txt  
  
Chocolate 50 300 chocolate

Caramel 10 50 caramel

Gummy 20 100 fruit

Waffle 30 200 wafer

Jelly 15 80 berry



