**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ КЫРГЫЗСКОЙ РЕСПУБЛИКИ**

**КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСТИТЕТ**

**им. И.Раззакова**

**ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

Кафедра «**Программное обеспечение компьютерных систем**»

Направление: 710400 «**Программная инженерия**»

Дисциплина: «**Объектно-ориентированное программирование**»

ОТЧЕТ

Лабораторная работа №2

Тема: “ **Классы и объекты** ”

Выполнила: Бактыбекова Айтунук

Группа: ПИ-2-21

Проверил: Мусабаев Э.Б.

Бишкек – 2024

Задание №1

Постановка задачи:

1. В здании аэропорта на экранах отображается информация о самолетах, а именно: информация о пункте отправления, пункте назначения, номере рейса, времени прибытия, времени отправления, номере секции для регистрации. Экраны – это средство, которое помогает своевременно зарегистрировать и отправить пассажиров. Важно, чтобы информация на экранах была понятной и верной.

Создайте необходимую информацию в виде таблицы для такого экрана, с помощью класса Aeroflot, содержащего в описании следующие поля: номер рейса; название пункта отправления; название пункта назначения рейса; время прибытия; время отправления; место регистрации.

Напишите код программы, выполняющей следующие действия: ввод с клавиатуры значений полей объектов; сортировку записей в таблице в алфавитном порядке по названию пунктов назначения; вывод на консоль значений полей объектов класса в виде таблицы рейсов; если таких рейсов нет, выдать соответствующее сообщение.

Описание исходных данных:

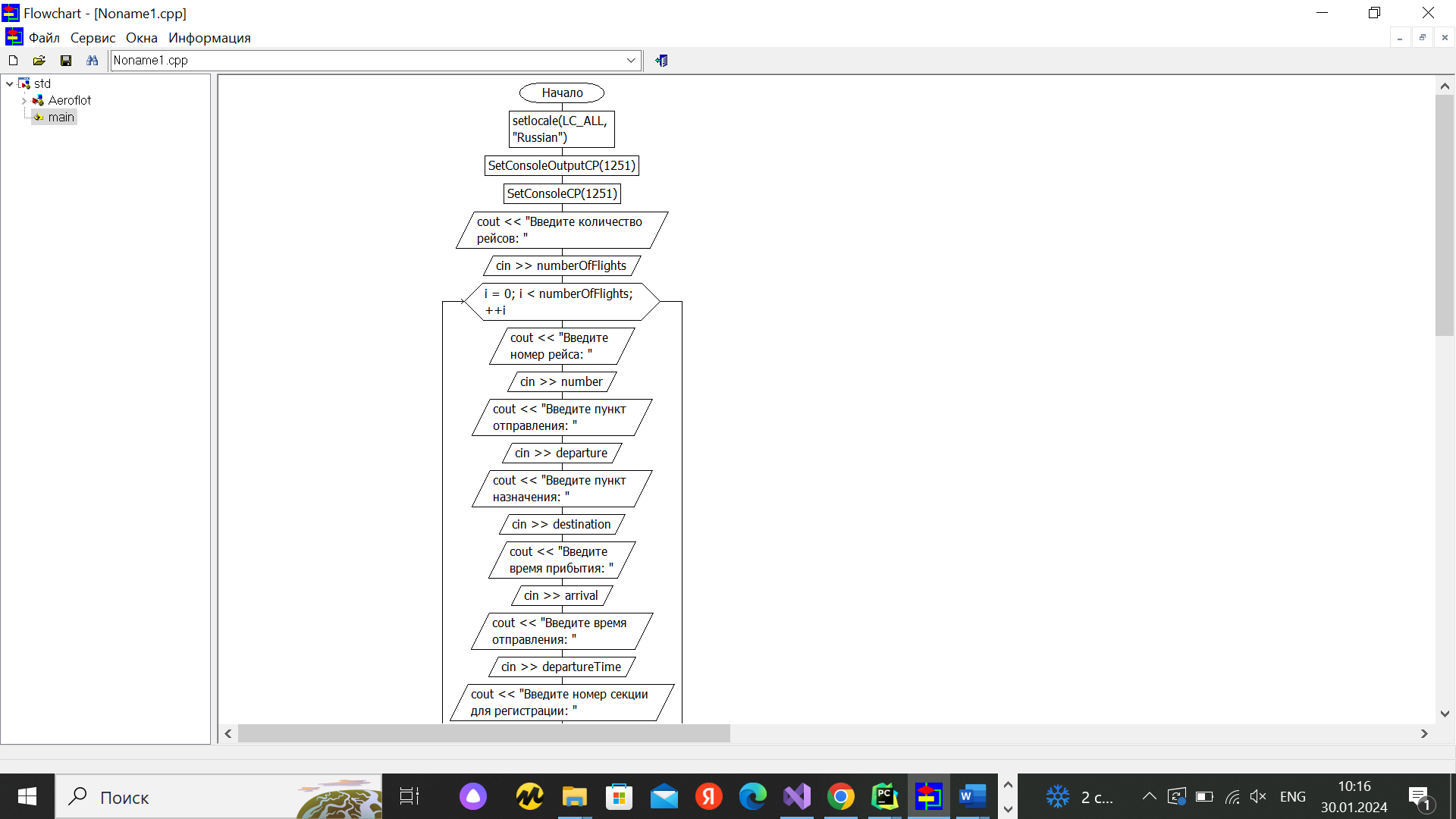
Класс Aeroflot()

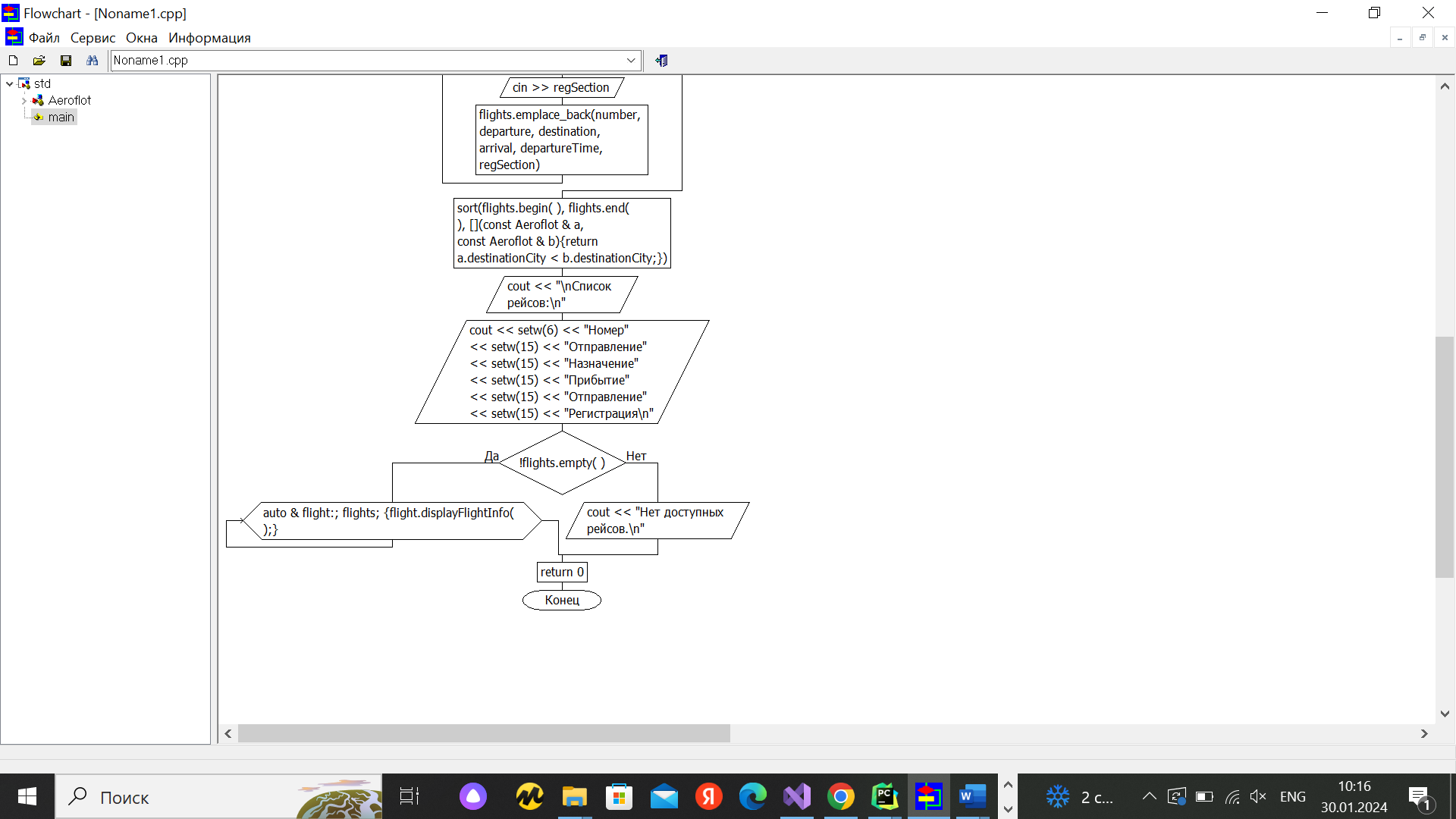
* **Параметры:**
  + **int flightNumber**: //номера рейса.
  + **string departureCity** //пункта отправления.
  + **string destinationCity //** пункта назначения
  + **string arrivalTime //** времени прибытия
  + **string departureTime //** времени отправления
  + **int registrationSection //** номера секции для регистрации

Конструктор класса **Aeroflot** инициализирует поля объекта при его создании. Метод **displayFlightInfo()** выводит информацию о рейсе в виде строки таблицы.

Далее запрашивается у пользователя количество рейсов и информацию о каждом рейсе (номер, пункт отправления, пункт назначения, время прибытия, время отправления, номер секции для регистрации), после чего создает объекты класса **Aeroflot** и добавляет их в вектор **flights**.

Блок-схема:





**Код программы:**

#include <iostream>

#include <iomanip>

#include <vector>

#include <algorithm>

#include <clocale>

#include <windows.h>

using namespace std;

class Aeroflot {

public:

int flightNumber;

string departureCity;

string destinationCity;

string arrivalTime;

string departureTime;

int registrationSection;

Aeroflot(int number, const string& departure, const string& destination,

const string& arrival, const string& departureTime, int registrationSection)

: flightNumber(number), departureCity(departure), destinationCity(destination),

arrivalTime(arrival), departureTime(departureTime), registrationSection(registrationSection) {}

void displayFlightInfo() const {

cout << setw(6) << flightNumber << setw(15) << departureCity << setw(15) << destinationCity

<< setw(15) << arrivalTime << setw(15) << departureTime << setw(15) << registrationSection << endl;

}

};

int main() {

setlocale(LC\_ALL, "Russian");

vector<Aeroflot> flights;

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

int numberOfFlights;

cout << "Введите количество рейсов: ";

cin >> numberOfFlights;

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

int number, regSection;

string departure, destination, arrival, departureTime;

cout << "Введите номер рейса: ";

cin >> number;

cout << "Введите пункт отправления: ";

cin >> departure;

cout << "Введите пункт назначения: ";

cin >> destination;

cout << "Введите время прибытия: ";

cin >> arrival;

cout << "Введите время отправления: ";

cin >> departureTime;

cout << "Введите номер секции для регистрации: ";

cin >> regSection;

flights.emplace\_back(number, departure, destination, arrival, departureTime, regSection);

}

sort(flights.begin(), flights.end(), [](const Aeroflot& a, const Aeroflot& b) {

if (a.departureCity != b.departureCity) {

return a.departureCity < b.departureCity;

}

return a.destinationCity < b.destinationCity;

});

cout << "\nСписок рейсов:\n";

cout << setw(6) << "Номер" << setw(15) << "Отправление" << setw(15) << "Назначение"

<< setw(15) << "Прибытие" << setw(15) << "Отправление" << setw(15) << "Регистрация\n";

if (!flights.empty()) {

for (const auto& flight : flights) {

flight.displayFlightInfo();

}

}

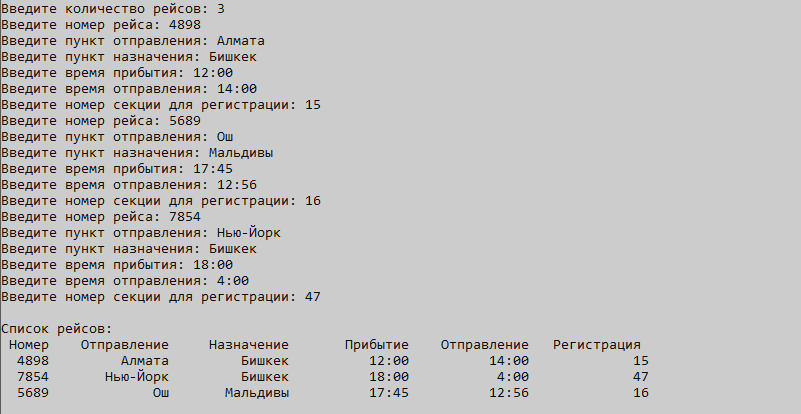
else {

cout << "Нет доступных рейсов.\n";

}

return 0;

}Результат:



Задание №2

Постановка задачи:

Предусмотрите возможность: работы с переменным числом студентов; поиска студента по какому-либо признаку (например, по фамилии, дате рождения или номеру телефона); добавления или удаления записей; сортировки по разным полям.

Разработайте программу, демонстрирующую работу с этим классом.

Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Описание исходных данных:

класс **Student**,

* **string lastName; //** фамилия
* **string dateOfBirth;//** даты рождения
* **string phoneNumber; //** номера телефона

Конструктор класса **Student** инициализирует поля объекта при его создании. Метод **displayInfo()** выводит информацию о студенте на экран.

У класса есть вектор объектов класса **Student**, который хранит информацию о студентах. Методы класса позволяют добавлять, удалять, находить студентов по фамилии, сортировать студентов по фамилии и отображать всех студентов.

В нем происходит инициализация объекта класса **StudentGroup**, установка кодировки консоли для корректного отображения кириллицы, а также создание меню для пользовательского ввода и выбора действий. С помощью оператора **switch** обрабатываются выбранные пользователем действия, такие как добавление, удаление, поиск, сортировка и отображение студентов. Программа выполняется до тех пор, пока пользователь не выберет выход из программы (ввод 0).

Код программы:

#include <iostream>

#include <vector>

#include <algorithm>

#include <windows.h>

using namespace std;

class Student {

public:

string lastName;

string dateOfBirth;

string phoneNumber;

Student(const string& last, const string& dob, const string& phone)

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

void displayInfo() const {

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

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

cout << "Телефон: " << phoneNumber << "\n";

}

};

class StudentGroup {

private:

vector<Student> students;

public:

void addStudent(const string& lastName, const string& dateOfBirth, const string& phoneNumber) {

students.emplace\_back(lastName, dateOfBirth, phoneNumber);

}

void removeStudent(const string& lastName) {

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

[&](const Student& student) { return student.lastName == lastName; }),

students.end());

}

Student\* findStudentByLastName(const string& lastName) {

auto it = find\_if(students.begin(), students.end(),

[&](const Student& student) { return student.lastName == lastName; });

return (it != students.end()) ? &(\*it) : nullptr;

}

void sortStudentsByLastName() {

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

[](const Student& a, const Student& b) { return a.lastName < b.lastName; });

}

void displayAllStudents() const {

if (students.empty()) {

cout << "Группа пуста." << endl;

}

else {

cout << "Список студентов:\n";

for (const auto& student : students) {

student.displayInfo();

cout << "\n";

}

}

}

};

int main() {

setlocale(LC\_ALL, "Russian");

StudentGroup group;

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

int choice;

do {

cout << "\nМеню:\n";

cout << "1. Добавить студента\n";

cout << "2. Удалить студента\n";

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

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

cout << "5. Показать всех студентов\n";

cout << "0. Выйти\n";

cout << "Выберите действие: ";

cin >> choice;

switch (choice) {

case 1: {

string lastName, dateOfBirth, phoneNumber;

cout << "Введите фамилию: ";

cin >> lastName;

cout << "Введите дату рождения: ";

cin >> dateOfBirth;

cout << "Введите номер телефона: ";

cin >> phoneNumber;

group.addStudent(lastName, dateOfBirth, phoneNumber);

break;

}

case 2: {

string lastName;

cout << "Введите фамилию студента, которого нужно удалить: ";

cin >> lastName;

group.removeStudent(lastName);

break;

}

case 3: {

string lastName;

cout << "Введите фамилию студента, которого нужно найти: ";

cin >> lastName;

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

if (foundStudent) {

cout << "Студент найден:\n";

foundStudent->displayInfo();

}

else {

cout << "Студент не найден.\n";

}

break;

}

case 4:

group.sortStudentsByLastName();

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

break;

case 5:

group.displayAllStudents();

break;

case 0:

cout << "Программа завершена.\n";

break;

default:

cout << "Неверный ввод. Повторите попытку.\n";

}

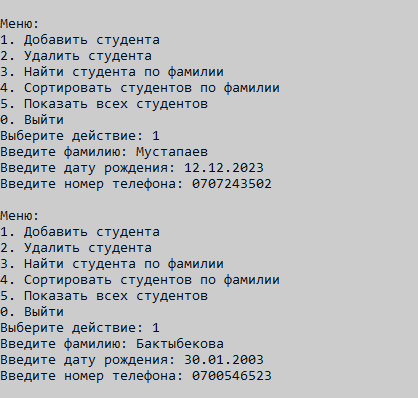
} while (choice != 0);

return 0;

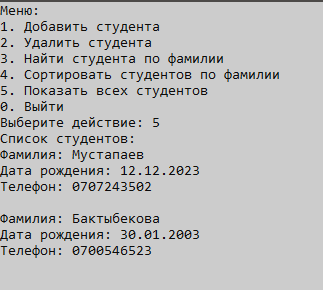
}

**Результат:**

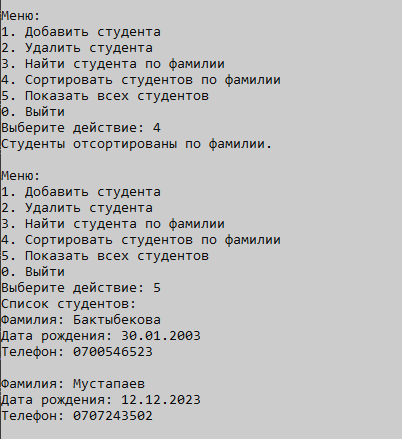
**Добавление студента**



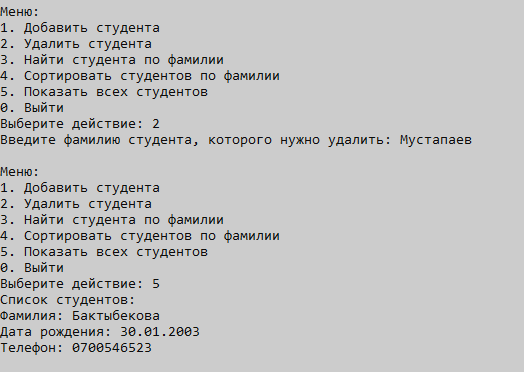
**Просмотр студентов**



**Сортировка студентов**



**Удаление студентов**



Задание №3

Постановка задачи:

Разработать программу, использующую этот класс для моделирования Т-образного сортировочного узла на железной дороге. Программа должна разделять на два направления состав, состоящий из вагонов двух типов (на каждое направление формируется состав из вагонов одного типа). Предусмотреть возможность формирования состава из файла или с клавиатуры. Возможно использование стандартных функций при работе со стеком в виде контейнера из библиотеки STL:

**push()** - поместить элемент в вершину стека;

**pop()** - удалить элемент из вершины стека;

класс **STACK**, который содержит четыре стека (**myStack1**, **myStack2**, **myStack3**, **myStack4**).

Класс имеет методы для ввода данных в стек, разделения вагонов на два стека по четности и нечетности, считывания данных из файла в стек, вывода содержимого стека, вывода четных и нечетных вагонов.

Происходит инициализация объекта класса **STACK**, генерация случайных чисел и запись их в файл, а также создание меню для пользовательского ввода и выбора действий. В зависимости от выбора пользователя вызываются соответствующие методы класса **STACK**.

Код программы:

#include <iostream>

#include <cstdlib>

#include <ctime>

#include <Windows.h>

#include <fstream>

#include<stack>

using namespace std;

class STACK {

private:

stack <int>myStack1;

stack <int>myStack2;

stack <int>myStack3;

stack <int>myStack4;

public:

void input(int a) {

myStack1.push(a);

myStack2 = myStack1;

}

void raz(int n) {

if (myStack1.empty()) {

cout << "Нет данных!" << endl;

}

else {

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

if (myStack1.top() % 2 == 0) {

myStack3.push(myStack1.top());

myStack1.pop();

}

else {

myStack4.push(myStack1.top());

myStack1.pop();

}

}

cout << "Деление завершено!" << endl;

}

}

void file\_Show() {

ifstream fin;

fin.open("file2.txt");

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

cout << "Файл не открылся! " << endl;

}

else {

int a;

while (fin >> a) {

myStack1.push(a);

}

myStack2 = myStack1;

}

fin.close();

cout << "Данные считаны из файла!!" << endl;

}

void Show() {

myStack2 = myStack1;

if (!myStack2.empty()) {

while (!myStack2.empty()) {

cout << myStack2.top() << endl;

myStack2.pop();

}

}

else {

cout << "Стек пуст!" << endl;

}

}

void chetn() {

myStack2 = myStack3;

if (!myStack2.empty()) {

while (!myStack2.empty()) {

cout << myStack2.top() << endl;

myStack2.pop();

}

}

else {

cout << "Нет таких вагонов!" << endl;

}

}

void ne\_chetn() {

myStack2 = myStack4;

if (!myStack2.empty()) {

while (!myStack2.empty()) {

cout << myStack2.top() << endl;

myStack2.pop();

}

}

else {

cout << "Нет таких вагонов!" << endl;

}

}

};

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

setlocale(LC\_ALL, "RUS");

srand(time(NULL));

int col = 0, zn = 0;

int val = 0;

cout << "Введите количество вагонов: ";

cin >> col;

ofstream fout;

STACK obj;

fout.open("file2.txt");

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

cout << "Файл не открылся!" << endl;

}

else {

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

fout << rand() % 20 << endl;

}

}

fout.close();

while (val != 6)

{

system("CLS");

cout << "\tМеню" << endl;

cout << "0) Ввод данных вагонов" << "\n1) Формирование состава из файла" << "\n2) Вывод " << "\n3) Разделить вагоны" << "\n4) Вывод четных вагонов" << "\n5) Вывод нечетных вагонов" <<"\n6) Выход" << endl;

cout << endl << "Выберите из меню: ";

cin >> val;

if (val == 0) {

system("CLS");

cout << "Введите номера вагонов (int): " << endl;

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

cin >> zn;

obj.input(zn);

}

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

system("pause");

}

else if (val == 1) {

system("CLS");

obj.file\_Show();

system("pause");

}

else if (val == 2) {

system("CLS");

cout << "\tВсе вагоны: " << endl << endl;

obj.Show();

system("pause");

}

else if (val == 3) {

system("CLS");

obj.raz(col);

system("pause");

}

else if (val == 4) {

system("CLS");

cout << "\tВагоны с четными номерами: " << endl << endl;

obj.chetn();

system("pause");

}

else if (val == 5) {

system("CLS");

cout << "\tВагоны с нечетными номерами: " << endl << endl;

obj.ne\_chetn();

system("pause");

}

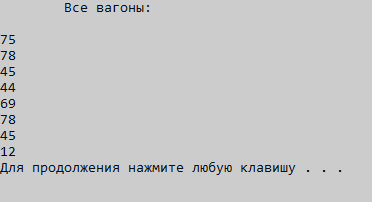
}

cout << "До свидания!" << endl;

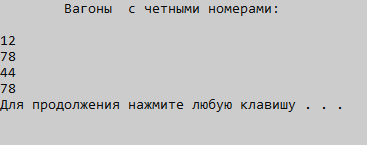
}

Результат:

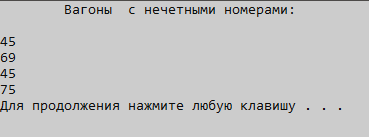
**Вывод всех вагонов**



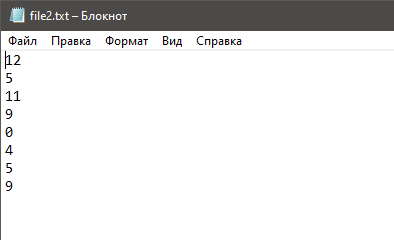
**Вывод вагонов с четными номерами**



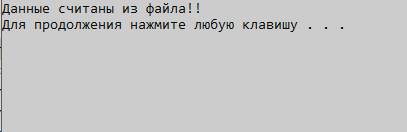
**Вывод вагонов с нечетными номерами**



**Файл**



**Считывание данных из файла**



**Вывод данных**

