Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа № 4

По дисциплине «Основы алгоритмизации и программирования»

На тему «Представление информации в виде структуры»

Выполнил:

Студент 1 курса 8 группы

Рауба Арсений Владимирович

Преподаватель: асс. Андронова М.В.

Минск, 2024

В соответствии со своим вариантом разработать программу для данных, приведенных в таблице ниже. Определить структурированный тип, разработать меню для работы с массивом структур. В программу должны войти функции: − ввод элементов структуры с клавиатуры; − вывод элементов структуры в консольное окно; − удаление заданной структурированной переменной; − поиск информации; − запись информации в файл; − чтение данных из файла.

**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

//структура для хранения информации о студенте

struct Student{

char name[10], secName[10], surname[10], specialization[30], faculty[50], date[8];

int group;

float averageScore;

};

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;

cout << "Введите имя:";

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

cout << "Введите отчество:";

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

cout << "Введите дату поступления:";

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

cout << "Введите факультет:";

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

cout << "Введите специальность:";

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

cout << "Введите группу:";

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

cout << "Введите средний балл:";

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

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

cout << "Дата поступления: ";

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

cout << "Факультет: ";

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

cout << "Специальность: ";

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

cout << "Группа: ";

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

cout << "Средний балл: ";

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

}

if (sw == 2)

{

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

{

cout << "ФИО: ";

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

cout << "Дата поступления: ";

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

cout << "Факультет: ";

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

cout << "Специальность: ";

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

cout << "Группа: ";

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

cout << "Средний балл: ";

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

}

}

}

void clear() {//функция для очистки

setlocale(LC\_ALL, "rus");

if (current\_size == 0)

{

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

return;

}

int delchoice;

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

<< "2 - удалить информацию о всех записанных студентах" << 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[0] = '\0';

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

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

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

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

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

list\_of\_student[i].group = 0;

list\_of\_student[i].averageScore = 0.0;

}

current\_size = 0;

cout << "Информация обо всех студентах 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 студента" << endl << endl;

return;

}

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

cin >> stud;

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

{

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

{

found = true;

cout << "ФИО: ";

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

cout << "Дата поступления: ";

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

cout << "Факультет: ";

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

cout << "Специальность: ";

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

cout << "Группа: ";

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

cout << "Средний балл: ";

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

break;

}

}

if (!found)

{

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

fout << "Дата поступления: ";

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

fout << "Факультет: ";

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

fout << "Специальность: ";

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

fout << "Группа: ";

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

fout << "Средний балл: ";

fout << list\_of\_student[i].averageScore << 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 студента" << endl;

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

cout << "3 - удалить введенных студентов" << 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:

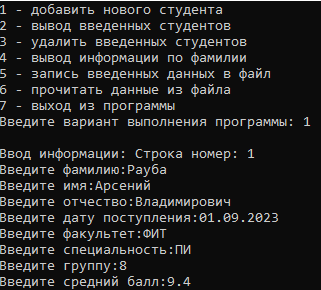
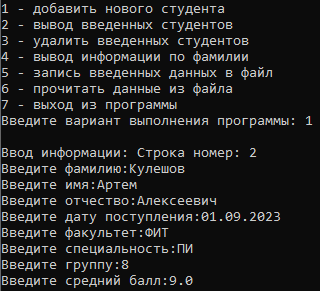
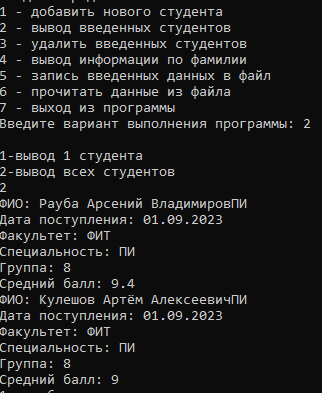
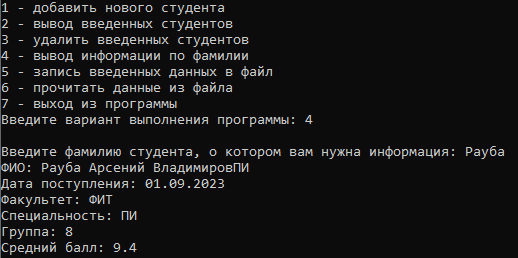
system("exit");

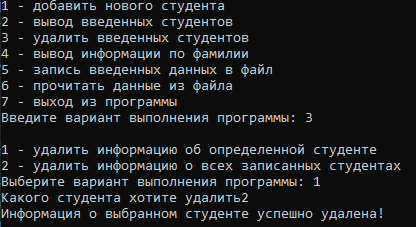
break;

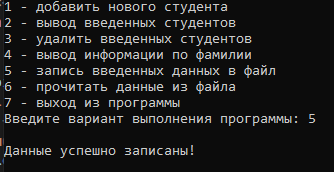
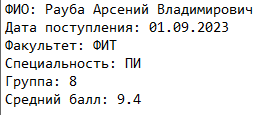
}

} while (choice != 7);

}

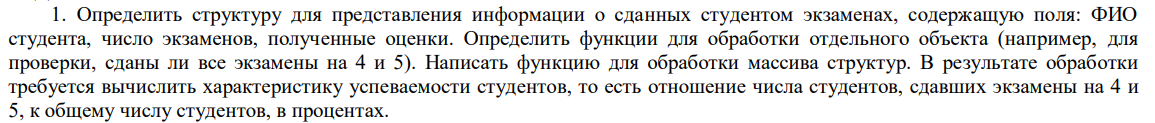
**   **

****

**** ****

**Доп. Задания**

**1.**

****

#include <iostream>

#include <fstream>

#include <Windows.h>

using namespace std;

const int MAX\_EXEM = 8;

const int MAX\_STUD\_COUNT = 10;

int studCount = 0;

struct Students

{

char FIO[40];

int count;

int marks[MAX\_EXEM];

};

Students stud\_list[MAX\_STUD\_COUNT];

bool otn[MAX\_STUD\_COUNT];

// Функция для ввода информации о студенте

void getInfo();

// Функция для проверки, сданы ли все экзамены определенного студента на 4 и 5

void isAll45();

// Функция для расчета характеристики успеваемости студентов

double stats();

int main()

{

// Установка кодировки для корректного отображения кириллицы в консоли

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int choice;

do

{

// Вывод меню и выбора пользователя

cout << "1 - добавить информацию о студенте" << endl

<< "2 - проверить, сданы ли все экзамены определенного студента на 4 и 5" << endl

<< "3 - вывести характеристику успеваемости студентов" << endl

<< "4 - выход" << endl

<< "Введите вариант выполнения программы: ";

cin >> choice;

// Обработка выбора пользователя

switch (choice)

{

case 1:

getInfo();

break;

case 2:

isAll45();

break;

case 3:

cout << "Успеваемость студентов = " << stats() << "%" << endl;

break;

case 4:

cout << "Спасибо за пользование нашей программой!" << endl;

system("pause");

}

} while (choice != 4);

}

void getInfo()

{

// Проверка на максимальное количество студентов

if (studCount == MAX\_STUD\_COUNT)

{

cout << "Вы ввели максимально допустимое число студентов!" << endl;

return;

}

Students newStud;

// Ввод информации о студенте

cout << "Введите ФИО студента: ";

cin.ignore();

cin.getline(newStud.FIO, 40);

do

{

cout << "Введите количество сданных экзаменов: ";

cin >> newStud.count;

if (newStud.count > MAX\_EXEM)

{

cout << "Студент не мог сдать такое количество экзаменов" << endl;

}

} while (newStud.count > MAX\_EXEM);

// Ввод оценок студента

for (int i = 0; i < newStud.count; i++)

{

do

{

cout << "Введите оценку под номером " << i + 1 << ": ";

cin >> newStud.marks[i];

if (newStud.marks[i] < 1 || newStud.marks[i] > 10)

{

cout << "Недопустимая отметка" << endl;

}

} while (newStud.marks[i] < 1 || newStud.marks[i] > 10);

}

// Добавление данных о студенте в список

stud\_list[studCount] = newStud;

studCount++;

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

<< endl;

}

void isAll45()

{

int ischoice;

bool isgood = true;

// Ввод номера студента для проверки оценок

cout << "Введите номер студента, оценки которого нужно проверить: ";

cin >> ischoice;

for (int i = 0; i < stud\_list[ischoice - 1].count; i++)

{

if (stud\_list[ischoice - 1].marks[i] != 9 && stud\_list[ischoice - 1].marks[i] != 10)

{

isgood = false;

break;

}

}

// Вывод результата проверки

if (isgood)

{

cout << "Студент сдал все экзамены на 9 и 10!" << endl

<< endl;

return;

}

if (!isgood)

{

cout << "Студент сдал экзамены с отметками ниже 9" << endl

<< endl;

return;

}

}

double stats()

{

// Подсчет количества студентов со всеми оценками 9 и 10

for (int k = 0; k < studCount; k++)

{

bool isgood = true;

for (int i = 0; i < stud\_list[k].count; i++)

{

if (stud\_list[k].marks[i] != 9 && stud\_list[k].marks[i] != 10)

{

isgood = false;

break;

}

}

otn[k] = isgood;

}

// Расчет процента студентов с отличной успеваемостью

double good = 0;

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

{

if (otn[i] == true)

{

good++;

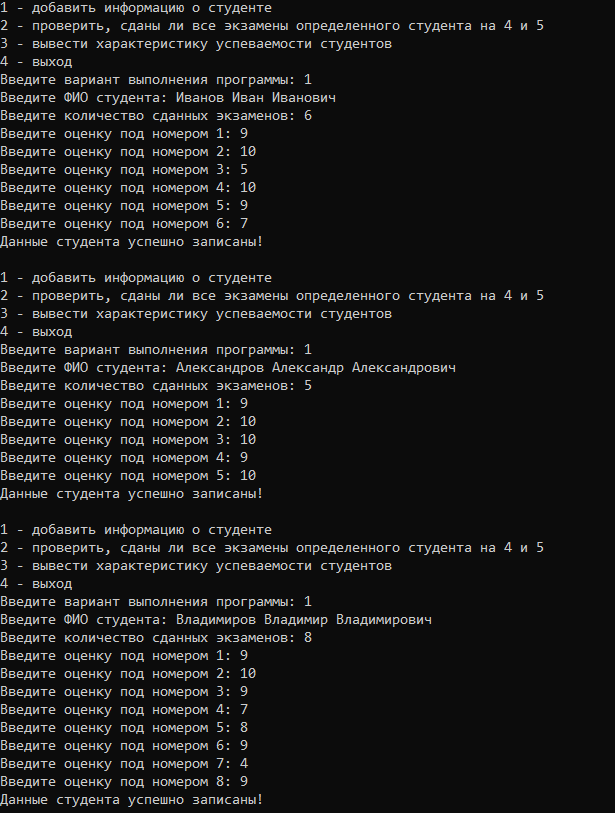
}

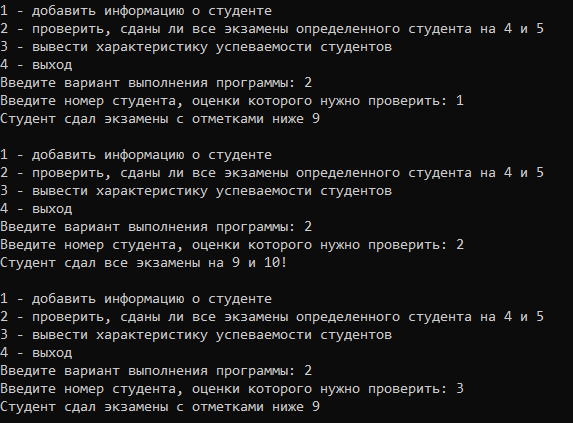
}

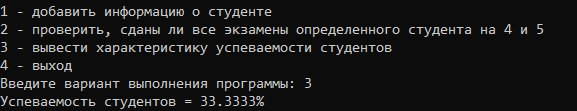
double attitude = (good / studCount) \* 100;

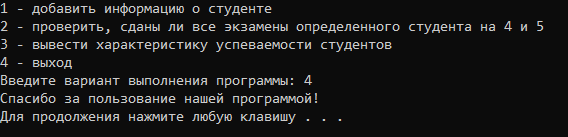
return attitude;

}

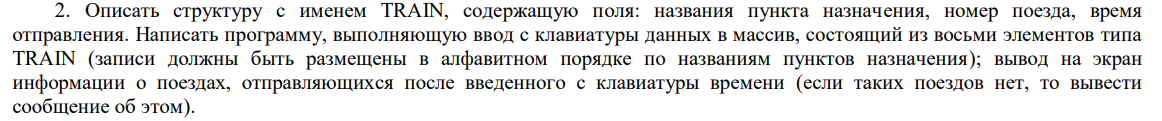
****

****

****

****

**2 проверяет, сданы ли все экзамены на 9 и 10, забыл поменять надпись**

**2.** ****

#include <iostream>

#include <Windows.h>

using namespace std;

const int MAX\_TRAINS\_COUNT = 8;

struct TRAIN

{

char name[20];

int number;

int time;

};

TRAIN trains\_list[MAX\_TRAINS\_COUNT];

void output();

#include <iostream>

#include <Windows.h>

using namespace std;

const int MAX\_TRAINS\_COUNT = 8;

int trainsCount = 0;

struct TRAIN

{

char name[20];

int number;

int time;

};

TRAIN trains\_list[MAX\_TRAINS\_COUNT];

void output();

int main()

{

// Установка кодировки консоли для поддержки кириллицы

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

// Ввод данных о поездах

TRAIN newTrain;

for (int i = 0; i < MAX\_TRAINS\_COUNT; i++)

{

cout << "Введите название пункта назначения: ";

cin >> newTrain.name;

cout << "Введите номер поезда: ";

cin >> newTrain.number;

cout << "Введите время отправления в формате ЧЧММ: ";

cin >> newTrain.time;

trains\_list[i] = newTrain;

}

// Сортировка по пункту назначения

for (int i = 0; i < MAX\_TRAINS\_COUNT - 1; i++)

{

for (int j = 0; j < MAX\_TRAINS\_COUNT - i - 1; j++)

{

if (strcmp(trains\_list[j].name, trains\_list[j + 1].name) > 0)

{

// Поменять местами поезда, если текущий больше следующего по названию пункта

TRAIN temp = trains\_list[j];

trains\_list[j] = trains\_list[j + 1];

trains\_list[j + 1] = temp;

}

}

}

// Вывод результатов

output();

}

void output()

{

int time, inhour, inminute, hour, minute;

bool found = false;

// Ввод времени и поиск поездов

cout << "Введите время в формате ЧЧММ, после которого нужно найти поезда: ";

cin >> time;

inminute = time % 100;

inhour = (time - inminute) / 100;

// Поиск и вывод поездов после указанного времени

for (int i = 0; i < MAX\_TRAINS\_COUNT; i++)

{

found = false;

minute = trains\_list[i].time % 100;

hour = (trains\_list[i].time - minute) / 100;

if (inhour < hour)

{

found = true; // Найден поезд, отправляющийся после указанного часа

}

if (inhour == hour && inminute < minute)

{

found = true; // Найден поезд, отправляющийся в указанное время

}

if (found)

{

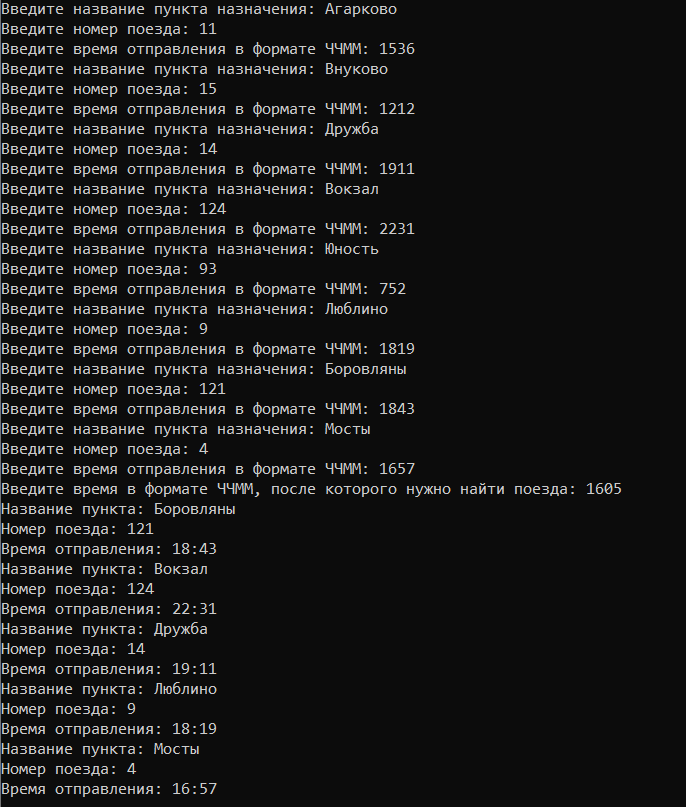
cout << "Название пункта: " << trains\_list[i].name << endl;

cout << "Номер поезда: " << trains\_list[i].number << endl;

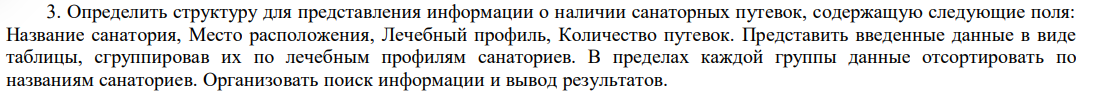
cout << "Время отправления: " << hour << ":" << minute << endl;

}

}



**3.**

****

#include <iostream>

#include <Windows.h>

#include <string>

#include <iomanip>

using namespace std;

const int MAX\_SANAT\_COUNT = 5;

void input();

void searchBySanatoriumName(string name);

string prof[5] = { "Болезнь нервной системы", "Болезнь органов пищеварения", "Кожные заболевания", "Заболевания органов дыхания нетуберкулезного характера", "Болезнь опорно-двигательного аппарата" };

struct Sanat

{

char name[20];

char place[20];

int prof;

int count;

};

Sanat sanat\_list[MAX\_SANAT\_COUNT];

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

input();

// Сортировка пузырьком по профилю и названию

for (int i = 0; i < MAX\_SANAT\_COUNT - 1; i++)

{

for (int j = 0; j < MAX\_SANAT\_COUNT - i - 1; j++)

{

if (sanat\_list[j].prof > sanat\_list[j + 1].prof || (sanat\_list[j].prof == sanat\_list[j + 1].prof && strcmp(sanat\_list[j].name, sanat\_list[j + 1].name) > 0)) {

swap(sanat\_list[j], sanat\_list[j + 1]);

}

}

}

// Вывод в виде таблицы, сгруппированной по профилю

cout << left << setw(20) << "Название санатория" << setw(20) << "Место расположения" << setw(40) << "Лечебный профиль" << "Количество путевок\n";

for (int i = 0; i < MAX\_SANAT\_COUNT; i++)

{

cout << left << setw(20) << sanat\_list[i].name << setw(20) << sanat\_list[i].place << setw(40) << prof[sanat\_list[i].prof - 1] << sanat\_list[i].count << "\n";

}

// Поиск по названию санатория

string searchName;

cout << "Введите название санатория для поиска: ";

cin >> searchName;

searchBySanatoriumName(searchName);

return 0;

}

void input()

{

Sanat newSanat;

cout << "Памятка по профилям санаториев: " <<

endl << "1 - Болезнь нервной системы" <<

endl << "2 - Болезнь органов пищеварения" <<

endl << "3 - Кожные заболевания" <<

endl << "4 - Заболевания органов дыхания нетуберкулезного характера" <<

endl << "5 - Болезнь опорно-двигательного аппарата" << endl;

for (int i = 0; i < MAX\_SANAT\_COUNT; i++)

{

cout << "Введите название санатория " << i + 1 << ":";

cin >> newSanat.name;

cout << "Введите место расположения санатория " << i + 1 << ":";

cin >> newSanat.place;

cout << "Введите номер лечебного профиля санатория " << i + 1 << ":";

cin >> newSanat.prof;

cout << "Введите количество путевок санатория " << i + 1 << ":";

cin >> newSanat.count;

sanat\_list[i] = newSanat;

}

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

}

void searchBySanatoriumName(string name)

{

bool found = false;

for (int i = 0; i < MAX\_SANAT\_COUNT; i++)

{

if (strcmp(sanat\_list[i].name, name.c\_str()) == 0)

{

cout << "Найден санаторий: " << sanat\_list[i].name << "\n";

found = true;

break;

}

}

if (!found)

{

cout << "Санаторий с таким названием не найден.\n";

}

}

