МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»
(БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №2

по дисциплине: Объектно-ориентированное программирование

тема: «Модульное программирование. Интерфейсы»

Выполнил: ст. группы ПВ-233

Ситников Алексей Павлович

Проверил:

Белгород 2025 г.

Цель работы: Получение навыков модульной декомпозиции предметной области, создания модулей. Разработка интерфейсов.

Код программы:

main:

```
#include <iostream>
#include <windows.h>
#include "classes.h"


int main(void) {
	SetConsoleCP(1251);
	SetConsoleOutputCP(1251);

	std::vector<Student> students;
	std::vector<Group> Groups;
	std::vector<teacher> teachers;

	std::cout << "Команды:\n1) Добавить нового ученика\n2) Удалить ученика\n3) Поставить оценку\n4) Добавить учителя\n5) Удалить учителя\n6) Вывести оценки ученика\n7) Вывести состав группы\n8) Поменять группу\n9) Создать группу\n10) Вывести группы\n11) Вывести учителей\nВведите 0 чтобы выйти";
	std::cout << "Введите команду\n";
	short comand;
	std::cin >> comand;
	while (comand != 0) {
		if (comand == 1) {
			std::cout << "Введите имя фамилию и id группы\n";
			std::string tempName;
			std::string tempSurname;
			int tempid;
			std::cin >> tempName >> tempSurname >> tempid;

			if (tempid >= id_group) {
				std::cout << "id не существует\n";
			}
			else{

				Student newStudent(tempName, tempSurname, &Groups[tempid]);
				students.push_back(newStudent);
				std::cout << "Ученик добавлен\n\n";
			}
		}
		else if (comand == 2) {
			std::cout << "Введите id ученика\n";
			int tempid;
			std::cin >> tempid;
			if (tempid >= id_student) {
				std::cout << "id не существует\n";
			}
			else {
				students[tempid].deleteStudent();

				students.erase(students.begin() + tempid);
				for (Student& temp : students) {
					if (temp.getId() > tempid) {
						temp.shiftStudents();
					}
				}
```

```
                        id_student--;
                        std::cout << "Ученик удалён\n\n";
                }
        }
        else if (comand == 3) {
                std::cout << "Введите id ученика, название предмета, его оценку
и id преподавателя\n";
                int tempidSt;
                std::string subject;
                short score;
                int tepidTc;
                std::cin >> tempidSt >> subject >> score >> tepidTc;
                if (tempidSt >= id_student || tepidTc >= id_teacher) {
                        std::cout << "id не существует\n";
                }
                else {
                        students[tempidSt].creatGrade(subject, score, tepidTc);
                        std::cout << "Оценка создана\n\n";
                }
        }
        else if (comand == 4) {
                std::cout << "Введите Имя фамилию учителя\n";
                std::string name;
                std::string surname;
                std::cin >> name >> surname;
                teacher newTeacher(name, surname);
                teachers.push_back(newTeacher);
                std::cout << "Учитель добавлен\n\n";
        }
        else if (comand == 5) {
                int tempid;
                std::cout << "Введите id учителя\n";
                std::cin >> tempid;
                if (tempid >= id_teacher) {
                        std::cout << "id не существует\n";
                }
                else {
                        teachers.erase(teachers.begin() + tempid);
                        for (teacher& temp : teachers) {
                                if (temp.id > tempid) {
                                        temp.shiftTeachers();
                                }
                        }
                        id_teacher--;
                        std::cout << "Учитель удалён\n\n";
                }
        }
        else if (comand == 6) {
                std::cout << "Введите id ученика\n";
                int tempidSt;
                std::cin >> tempidSt;

                if (tempidSt >= id_student) {
                        std::cout << "id не существует\n";
                }
                else {
                        students[tempidSt].printGrade();
                }
        }
        else if (comand == 7) {
                std::cout << "Введите id группы\n";
                int tempid;
                std::cin >> tempid;

                if (tempid >= id_group) {
```

```
                        std::cout << "id не существует\n";
                }
                else {
                        Groups[tempid].printStudents();
                }
        }
        else if (comand == 8) {
                std::cout << "Введите id ученика и новой группы\n";
                int tempidSt;
                int tempidGr;
                std::cin >> tempidSt >> tempidGr;
                int lastid = students[tempidSt].getGroup()->getId();
                if (tempidGr >= id_group || tempidSt >= id_student) {
                        std::cout << "id не существует\n";
                }
                else {
                        students[tempidSt].changeGroup(&Groups[tempidGr]);
                        std::cout << "Ученик " + students[tempidSt].getFio() + "
переведён из группы: " + Groups[lastid].getName() + " в группу " +
Groups[tempidGr].getName() + "\n\n";
                }
        }
        else if (comand == 9) {
                std::cout << "Введите имя группы\n";
                std::string name;
                std::cin.ignore();
                std::getline(std::cin, name);
                Group newGroup(name);
                Groups.push_back(newGroup);
                std::cout << "Группа создана\n\n";
        }
        else if (comand == 10) {
                for (Group temp : Groups) {
                        temp.printGroup();
                }
        }
        else if (comand == 11) {
                std::cout << "Учителя:\n";
                for (teacher temp : teachers) {
                        temp.printTeacher();
                }
                std::cout << std::endl;
        }
        std::cout << "\nВведите команду\n";
        std::cin >> comand;
    }
    system("pause");
    return 0;
}
```

classes.h

```
#include <string>
#include <unordered_map>
#include <vector>
#include <map>


extern int id_group;
extern int id_teacher;
extern int id_student;

typedef struct grade {
    private:
        int id_teacher;
```

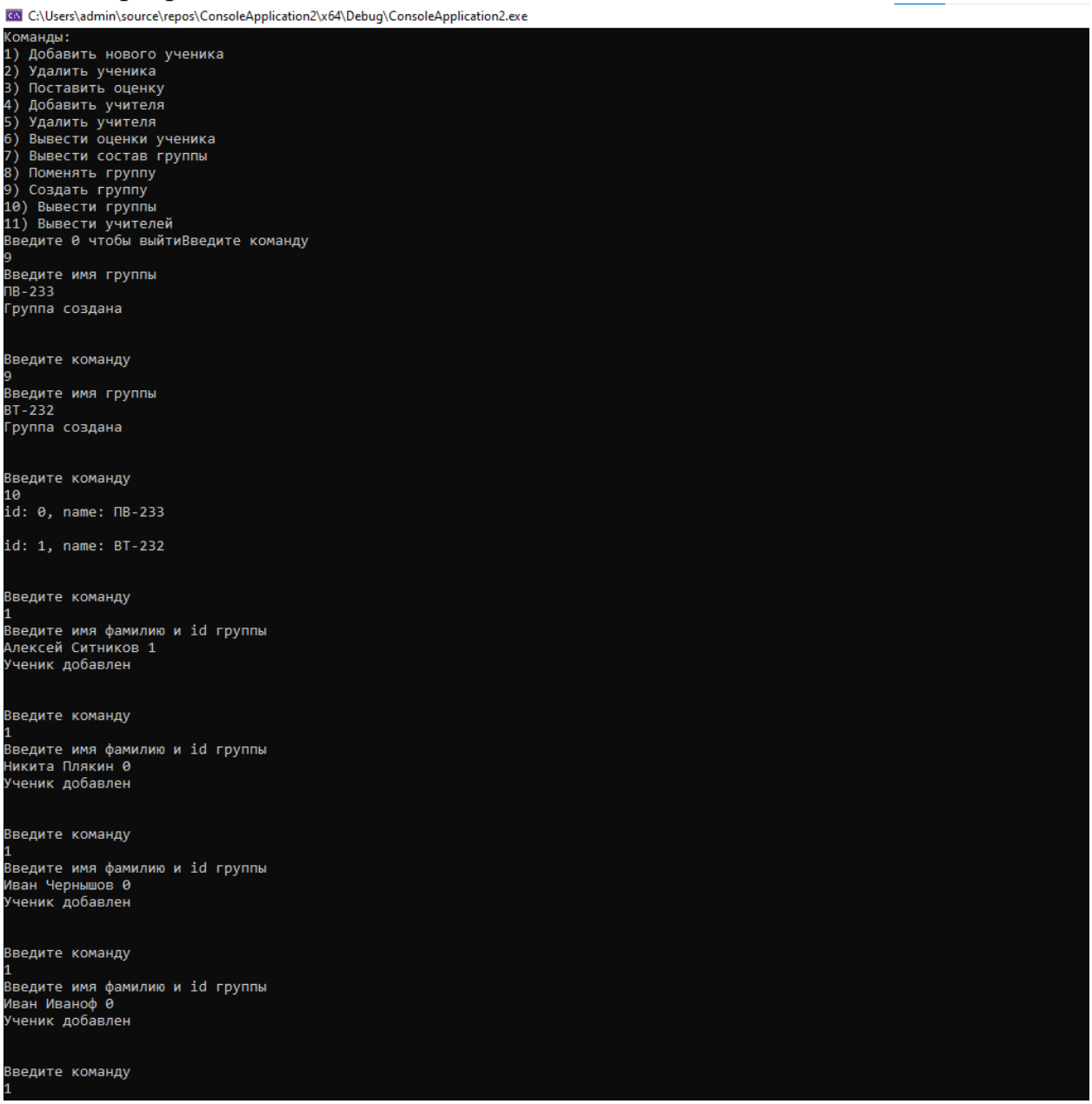
```
		std::string subject;
		short score;
	public:
		grade(std::string subject_, short score_, int id_teacher_);
		short getScore();

}grade;

typedef struct grades {
	private:
		std::map<std::string, std::vector<grade>> data;
	public:
		void creatGrade(std::string subject, short score, int id_teacher);
		void printGrade();
}grades;

typedef struct Group {
	private:
		std::vector<std::string> students;
		int id;
		std::string name;
	public:
		int getId();
		std::string getName();
		std::vector<std::string>* getStudents();
		Group(std::string name_);
		void printStudents();
		void printGroup();
		void deleteStudent(std::string str);
}Group;

typedef struct Student{
	private:
		std::string name;
		std::string surname;
		grades data;
		int id;
		Group* group;

	public:
		std::string getFio();
		int getId();
		Group* getGroup();
		Student(std::string name_, std::string surname_, Group* group_);
		void creatGrade(std::string subject, short score, int id_teacher);
	void printGrade();
	void changeGroup(Group* newGroup);
	void shiftStudents();
	void deleteStudent();


}student;

typedef struct teacher {
	int id;
	std::string name;
	std::string surname;
	teacher(std::string name_, std::string surname_);
	void shiftTeachers();
	void printTeacher();
}teacher;
```

classes.cpp

```cpp
#include "classes.h"
#include <iostream>
#include <algorithm>


int id_group = 0;
int id_teacher = 0;
int id_student = 0;

grade::grade(std::string subject_, short score_, int id_teacher_) {
	score = score_;
	id_teacher = id_teacher_;
	subject = subject_;
}

short grade::getScore() {
	return score;
}

void grades::creatGrade(std::string subject, short score, int id_teacher) {
	score = score < 2 ? 2 : score;
	score = score > 5 ? 5 : score;
	grade temp(subject, score, id_teacher);
	data[subject].push_back(temp);
}
void grades::printGrade() {
	for (auto& tempData : data) {
		std::cout << tempData.first << ": ";
		for (grade& tempGrades : tempData.second) {
			std::cout << tempGrades.getScore() << ' ';
		}
		std::cout << std::endl;
	}
}

int Group::getId() {
	return id;
}
std::string Group::getName() {
	return name;
}
std::vector<std::string>* Group::getStudents() {
	return &students;
}
Group::Group(std::string name_) {
	name = name_;
	id = id_group++;
}
void Group::printStudents() {
	std::sort(students.begin(), students.end());
	std::cout << "Студенты группы " + name + ':' << std::endl;
	for (std::string& student : students) {
		std::cout << student << std::endl;
	}
	std::cout << std::endl;
}
void Group::printGroup() {
	std::cout << "id: " << id << ", name: " << name << std::endl << std::endl;
}
void Group::deleteStudent(std::string str) {
	students.erase(std::remove(students.begin(), students.end(), str),
students.end());
}
```

```cpp
std::string Student::getFio() {
	return surname + ' ' + name;
}
int Student::getId() {
	return id;
}
Group* Student::getGroup() {
	return group;
}
Student::Student(std::string name_, std::string surname_, Group* group_) {
	name = name_;
	surname = surname_;
	group = group_;
	id = id_student++;
	std::string tempFio = surname_ + ' ' + name_ + ' ' + std::to_string(id);
	group->getStudents()->push_back(tempFio);
}
void Student::creatGrade(std::string subject, short score, int id_teacher) {
	data.creatGrade(subject, score, id_teacher);
}
void Student::printGrade() {
	std::cout << "Оценки ученика: " + surname + " " + name << std::endl;
	data.printGrade();
}
void Student::changeGroup(Group* newGroup) {
	std::string tempFio = surname + ' ' + name + ' ' + std::to_string(id);
	group->getStudents()->erase(std::remove(group->getStudents()->begin(), group-
>getStudents()->end(), tempFio), group->getStudents()->end());
	group = newGroup;
	group->getStudents()->push_back(tempFio);
}
void Student::shiftStudents() {
	std::string tempFio = surname + ' ' + name + ' ' + std::to_string(id);
	group->getStudents()->erase(std::remove(group->getStudents()->begin(), group-
>getStudents()->end(), tempFio), group->getStudents()->end());
	id--;
	tempFio = surname + ' ' + name + ' ' + std::to_string(id);
	group->getStudents()->push_back(tempFio);
}
void Student::deleteStudent() {
	std::string tempFio = surname + ' ' + name + ' ' + std::to_string(id);
	group->deleteStudent(tempFio);
}

teacher::teacher(std::string name_, std::string surname_) {
	id = id_teacher++;
	name = name_;
	surname = surname_;
	std::string tempFio = name_ + ' ' + surname_;
}
void teacher::shiftTeachers() {
	id--;
}
void teacher::printTeacher() {
	std::cout << id << " " + surname + " " + name + '\n';
}
```

Вывод программы:



C:\Users\admin\source\repos\ConsoleApplication2\x64\Debug\ConsoleApplication2.exe

```
Введите команду
1
Введите имя фамилию и id группы
Иван Иванов 0
Ученик добавлен


Введите команду
7
Введите id группы
0
Студенты группы ПВ-233:
Иванов Иван 4
Иваноф Иван 3
Плякин Никита 1
Чернышов Иван 2


Введите команду
2
Введите id ученика
3
Ученик удалён


Введите команду
7
Введите id группы
0
Студенты группы ПВ-233:
Иванов Иван 3
Плякин Никита 1
Чернышов Иван 2


Введите команду
8
Введите id ученика и новой группы
0 0
Ученик Ситников Алексей переведён из группы: ВТ-232 в группу ПВ-233


Введите команду
7
Введите id группы
1
Студенты группы ВТ-232:


Введите команду
7
Введите id группы
0
Студенты группы ПВ-233:
Иванов Иван 3
Плякин Никита 1
Ситников Алексей 0
Чернышов Иван 2


Введите команду
4
```

C:\Users\admin\source\repos\ConsoleApplication2\x64\Debug\ConsoleApplication2.exe

```
Введите команду
4
Введите Имя фамилию учителя
Дмитрий Буханов
Учитель добавлен


Введите команду
4
Введите Имя фамилию учителя
Андрей Хлопов
Учитель добавлен


Введите команду
11
Учителя:
0 Буханов Дмитрий
1 Хлопов Андрей


Введите команду
3
Введите id ученика, название предмета, его оценку и id преподавателя
0 ООП 4 0
Оценка создана


Введите команду
3
Введите id ученика, название предмета, его оценку и id преподавателя
0 ООП 5 0
Оценка создана


Введите команду
3
Введите id ученика, название предмета, его оценку и id преподавателя
0 алгебра 4 1
Оценка создана


Введите команду
3
Введите id ученика, название предмета, его оценку и id преподавателя
0 алгебра 3 1
Оценка создана


Введите команду
6
Введите id ученика
0
Оценки ученика: Ситников Алексей
ООП: 4 5
алгебра: 4 3

Введите команду
7
Введите id группы
0
Студенты группы ПВ-233:
```

```
Студенты группы ПВ-233:
Иванов Иван 3
Плякин Никита 1
Ситников Алексей 0
Чернышов Иван 2


Введите команду
2
Введите id ученика
0
Ученик удалён


Введите команду
7
Введите id группы
1
Студенты группы ВТ-232:


Введите команду
7
Введите id группы
0
Студенты группы ПВ-233:
Иванов Иван 2
Плякин Никита 0
Чернышов Иван 1


Введите команду
11
Учителя:
0 Буханов Дмитрий
1 Хлопов Андрей


Введите команду
5
Введите id учителя
0
Учитель удалён


Введите команду
11
Учителя:
0 Хлопов Андрей


Введите команду
0
Для продолжения нажмите любую клавишу . . .
```

Вывод: в ходе выполнения лабораторной работы были получены навыки модульной декомпозиции предметной области, создания модулей и разработки интерфейсов. Программа была успешно разделена на три модуля, что позволило сделать её более гибкой, расширяемой и поддерживаемой. Использование интерфейсов обеспечило абстракцию и упростило взаимодействие между модулями.