МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ» (ДГТУ)**

Факультет
Информатика и вычислительная техника

Кафедра
Программное обеспечение вычислительной техники и автоматизированных систем

# ОТНОШЕНИЕ МЕЖДУ КЛАССАМИ: АГРЕГАЦИЯ И КОМПОЗИЦИЯ

## Практикум

по выполнению лабораторной работы №4
по дисциплине «Программирование»

Программное обеспечение вычислительной техники и автоматизированных систем
*(указывается направленность (профиль) образовательной программы)*

09.03.04 Программная инженерия
*(указывается код и наименование направления подготовки)*

Ростов-на-Дону
2024 г.

Составители: канд. техн. наук, доц. каф. А.А. Скляренко

УДК 004.432

Отношение между классами Ассоциация: метод. указания. – Ростов н/Д:
Издательский центр ДГТУ, 2024.

В методическом указании рассматриваются вопросы отношений между классами, их виды и особенности. Приведены задания к лабораторной работе, помогающие закрепить на практике полученные знания, и контрольные вопросы для самопроверки. Предназначено для обучающихся по направлению 09.03.04 «Программная инженерия», профиль «Программное обеспечение вычислительной техники и автоматизированных систем».

Ответственный за выпуск:
зав. кафедрой «Программное обеспечение вычислительной техники и автоматизированных систем» Долгов В.В.

# 1 Теоретическая часть

## 1.1 Отношения между классами

В объектно-ориентированном программировании (ООП) отношения между классами определяют, как объекты одного класса взаимодействуют с объектами других классов. Это взаимодействие может быть основано на обмене данными, взаимодействии или иерархии.

Виды отношений между классами:

1. Ассоциация: Связь между объектами, не предполагающая владения или контроля. Отношение "один к одному", "один ко многим" или "многие ко многим". Реализуется с помощью ссылок на объекты.
2. Агрегация: Отношение "целое-часть", где один объект (целое) содержит и использует другие объекты (части), но не управляет их жизненным циклом. Части могут существовать независимо от целого.
3. Композиция: Сильный вариант агрегации, где части являются неотъемлемой частью целого и не могут существовать самостоятельно после его разрушения.

Ассоциация – это когда один класс включает в себя другой класс в качестве одного из полей. Ассоциация описывается словом «имеет». Автомобиль имеет двигатель. Вполне естественно, что он не будет являться наследником двигателя (хотя такая архитектура тоже возможна в некоторых ситуациях).

Выделяют два частных случая ассоциации: композицию и агрегацию.

Композиция – это когда двигатель не существует отдельно от автомобиля. Он создается при создании автомобиля и полностью управляется автомобилем. В типичном примере, экземпляр двигателя будет создаваться в конструкторе автомобиля.

```
class Engine
{
    int power;
    public Engine(int p)
    {
        power = p;
    }
}

class Car
{
    string model = "Porshe";
    Engine engine;
    public Car()
    {
        this.engine = new Engine(360);
    }
}
```

Агрегация – это когда экземпляр двигателя создается где-то в другом месте кода, и передается в конструктор автомобиля в качестве параметра.

```
class Engine
{
    int power;
    public Engine(int p)
    {
       power = p;
    }
}

class Car
{
    string model = "Porshe";
    Engine engine;
    public Car(Engine someEngine)
    {
        this.engine = someEngine;
    }
}

Engine goodEngine = new Engine(360);
Car porshe = new Car(goodEngine);
```

## 2 Задание к лабораторной работе

Разработать программу на тему «Отношение между классами (ассоциация, агрегация и композиция)», реализовав один из вариантов задания в соответствии с таблицей. Вариант согласовать с преподавателем.

Таблица 1 – Задание №1 «Отношение между классами»

| № вар-та | Задание |
|---|---|
| 1 | Онлайн-Каталог Книг<br>1. Создайте класс `Book`, который содержит информацию о книге: название, автор, ISBN, год издания.<br>- Агрегация: Создайте класс `Publisher`, который агрегирует в себе несколько объектов `Book`.<br>- Композиция: В классе `Book` создайте композицию класса `Cover`, содержащего информацию о обложке книги.<br>- Ассоциация: Объект `Book` связан с классом `Author`, содержащим информацию о авторе книги.<br><br>2. Создайте класс `Library`, который включает коллекцию объектов `Book`. Реализуйте методы для вывода списка всех книг, поиска по автору и по году издания. Напишите демонстрационную программу, использующую все методы классов. |
| 2 | Система Учёта Учащихся<br>1. Создайте класс `Student`, с полями имя, фамилия, ID и коллекцией оценок.<br>- Агрегация: Создайте класс `Classroom`, агрегирующий несколько объектов `Student`.<br>- Композиция: В классе `Student` используйте класс `ContactInfo` для хранения контактной информации ученика.<br>- Ассоциация: Объект `Student` ассоциируется с классом `Teacher`, представляющим учителя, который ведет у ученика занятия.<br><br>2. Создайте класс `School`, содержащий массив объектов `Classroom`. Реализуйте метод для вывода списка всех учеников и их оценок. Напишите демонстрационную программу, использующую все методы классов. |

| № вар-та | Задание |
|---|---|
| 3 | Автомобильный Сервис<br>1. Создайте класс `Car`, содержащий поля марка, модель, год выпуска, пробег.<br>- Агрегация: Создайте класс `Garage`, который агрегирует несколько объектов `Car`.<br>- Композиция: В классе `Car` используйте класс `Engine`, представляющий информацию о двигателе.<br>- Ассоциация: Объект `Car` ассоциируется с классом `Owner`, который содержит информацию о владельце автомобиля.<br><br>2. Создайте класс `ServiceStation`, который включает массив объектов `Garage`. Реализуйте методы для проведения техосмотра и задния стоимости ремонта для каждого автомобиля. Напишите демонстрационную программу, использующую все методы классов. |
| 4 | Музейная Экспозиция<br>1. Создайте класс `Artwork`, с полями название, автор, год создания, описание.<br>- Агрегация: Создайте класс `Exhibition`, который агрегирует несколько объектов `Artwork`.<br>- Композиция: В классе `Artwork` используйте класс `Dimensions`, для хранения размеров произведения.<br>- Ассоциация: Объект `Artwork` ассоциируется с классом `Gallery`, представляющим галерею, в которой хранится произведение.<br><br>2. Создайте класс `Museum`, содержащий массив объектов `Exhibition`. Реализуйте методы для вывода всех произведений искусства, их авторов и информации о текущих выставках. Напишите демонстрационную программу, использующую все методы классов. |
| 5 | Учебный Портал<br>1. Создайте класс `Course`, с полями название, описание, длительность, преподаватель.<br>- Агрегация: Создайте класс `Curriculum`, который агрегирует несколько объектов `Course`.<br>- Композиция: В классе `Course` используйте класс `Syllabus`, который хранит информацию о учебной программе курса.<br>- Ассоциация: Объект `Course` ассоциируется с классом `Student`, представляющим студентов, записанных на курс.<br><br>2. Создайте класс `EducationalPlatform`, содержащий массив объектов `Curriculum`. Реализуйте методы для вывода всех курсов, поиска по преподавателю и студентам, записанным на курсы. Напишите демонстрационную программу, использующую все методы классов. |

| № вар-та | Задание |
|---|---|
| 6 | Управление Пациентами в Больнице1. Создайте класс `Patient`, с полями ФИО, возраст, история болезни.<br>- Агрегация: Создайте класс `Department`, агрегирующий несколько объектов `Patient`.<br>- Композиция: В классе `Patient` используйте класс `MedicalRecord`, для хранения информации о медицинских записях пациента.<br>- Ассоциация: Объект `Patient` ассоциируется с классом `Doctor`, представляющим лечащего врача пациента.<br><br>2. Создайте класс `Hospital`, содержащий массив объектов `Department`. Реализуйте методы для управления пациентами и врачами, а также для просмотра истории болезней пациентов. Напишите демонстрационную программу, использующую все методы классов. |
| 7 | Логистика и Управление Транспортом<br>1. Создайте класс `Truck`, с полями марка, модель, грузоподъемность, текущий груз.<br>- Агрегация: Создайте класс `LogisticsCompany`, агрегирующий несколько объектов `Truck`.<br>- Композиция: В классе `Truck` используйте класс `Cargo`, представляющий информацию о грузе, перевозимом на текущий момент.<br>- Ассоциация: Объект `Truck` ассоциируется с классом `Driver`, представляющим водителя грузовика.<br><br>2. Создайте класс `FleetManager`, содержащий массив объектов `LogisticsCompany`. Реализуйте методы для управления грузовиками и водителями, а также для отслеживания перевозок. Напишите демонстрационную программу, использующую все методы классов. |
| 8 | Управление Авиаперелетами<br>1. Создайте класс `Flight`, с полями рейс, пункт отправления, пункт назначения, время отправления и прибытия.<br>- Агрегация: Создайте класс `Airline`, который агрегирует несколько объектов `Flight`.<br>- Композиция: В классе `Flight` используйте класс `SeatMap`, содержащий информацию о местах в самолете.<br>- Ассоциация: Объект `Flight` ассоциируется с классом `Pilot`, представляющим пилота, ведущего рейс.<br><br>2. Создайте класс `Airport`, содержащий коллекцию объектов `Airline`. Реализуйте методы для отображения расписания рейсов, поиска по авиалиниям и информации о пилотах. Напишите демонстрационную программу, использующую все методы классов. |

| № вар-та | Задание |
|---|---|
| 9 | Управление Торговым Центром<br>1. Создайте класс `Store`, с полями название, тип магазина, часы работы.<br>- Агрегация: Создайте класс `Mall`, который агрегирует несколько объектов `Store`.<br>- Композиция: В классе `Store` используйте класс `Inventory`, представляющий ассортимент товаров.<br>- Ассоциация: Объект `Store` ассоциируется с классом `Employee`, представляющим работников магазина.<br><br>2. Создайте класс `MallManager`, содержащий массив объектов `Mall`. Реализуйте методы для управления магазинами и сотрудниками, а также для учета и инвентаризации товара. Напишите демонстрационную программу, использующую все методы классов. |
| 10 | Управление Фермерским Хозяйством<br>1. Создайте класс `Plant`, с полями вид, стадия роста, время посадки.<br>- Агрегация: Создайте класс `Field`, который агрегирует несколько объектов `Plant`.<br>- Композиция: В классе `Plant` используйте класс `GrowthRecord`, содержащий информацию о росте растения.<br>- Ассоциация: Объект `Plant` ассоциируется с классом `Farmer`, представляющим фермеров, ухаживающих за растениями.<br><br>2. Создайте класс `Farm`, содержащий массив объектов `Field`. Реализуйте методы для управления посадками, учета работы фермеров и мониторинга роста растений. Напишите демонстрационную программу, использующую все методы классов. |

# 3 Материально-техническое обеспечение работы

Аудитория для проведения лабораторных занятий должна быть укомплектована специализированной мебелью и индивидуальными компьютерами следующей минимальной комплектации:

- Процессор: не менее двух исполнительных ядер, совместимый с системой команд x86 и x64, с поддержкой аппаратной виртуализации.
- Оперативная память: не менее 8 Гб.
- Монитор: не менее 24" (дюймов) по диагонали.
- Наличие локальной сети со скоростью обмена не менее 1 Гб/сек.
- Наличие доступа в сеть Интернет со скоростью не менее 1 Мбит/сек.
- Наличие клавиатуры и манипулятора «мышь».

На компьютерах должно быть установлено следующее программное обеспечение:

– Операционная система: Microsoft Windows 10 (или выше) и/или одна из операционных систем семейства Linux, допускающих установку среды исполнения для C++. Допускается конфигурация, когда одна из операционных систем установлена внутри виртуальной машины (гипервизора).
– Среды разработки и исполнения программ для C++.

## 4 Порядок выполнения и сдачи работы

Для выполнения лабораторной работы рекомендуется придерживаться следующего порядка выполнения:

1. Ознакомится с темой и целями лабораторной работы
2. Изучить теоретический материал
3. Подготовить рабочее окружение
4. Разработать программный код лабораторной работы в соответствие с заданием
5. Произвести проверку работоспособности, выполнить тестирование и отладку кода
6. Проанализировать полученные результаты и примененные в ходе решения подходы
7. Выполнить самооценку и рефлексию
8. Сдать лабораторную работу преподавателю и получить от него обратную связь

При сдаче студентом лабораторной работы основным отчетом выступает исходный код самостоятельно созданной в процессе выполнения работы программы.

Исходный код должен быть отформатирован согласно принятым для используемого языка программирования стандартам. Является желательным наличие в исходном коде комментариев, описывающих основные части программы и особенности их функционирования. В то же время студент должен быть готов объяснить работу программы в целом и каждую отдельную ее часть при полном отсутствии комментариев (например, они могут быть удалены преподавателем при сдаче работы).

Обязательным условием сдачи является умение студента самостоятельно восстановить любой участок исходного кода программы (но не более 20 строк подряд) после его удаления. Удаленный участок должен быть самостоятельно набран

студентом заново в присутствие преподавателя. При восстановлении участка запрещается использовать операции Undo (Отменить) текстовых редакторов или переписывание кода участка из других источников.

Исходный код сдаваемой программы должен быть представлен в электронном виде.

Сдача исходного кода ранее сдававшихся программ или программ, код которых выложен в сети Интернет, не допускается.

## 5 Контрольные вопросы к лабораторной работе

1. Опишите основные типы отношений между классами: ассоциация, агрегация и композиция.
2. В чем заключаются ключевые отличия между этими типами отношений?
3. Приведите примеры из реального мира для каждого типа отношения.
4. Какие существуют способы реализации этих отношений в C++?
5. Какие еще механизмы взаимодействия между классами существуют в ООП?

## 6 Перечень использованных информационных ресурсов

1. Ноткин, А. М. Объектно-ориентированное программирование: ООП на языке C++ : учебное пособие / А. М. Ноткин. — Пермь : ПНИПУ, 2013. — 230 с. — ISBN 978-5-398-00966-8. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/160806 (дата обращения: 22.08.2024). — Режим доступа: для авториз. пользователей.

2. Скворцова, Л. А. Объектно-ориентированное программирование на языке C++ : учебное пособие / Л. А. Скворцова. — Москва : РТУ МИРЭА, 2020. — 246 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/163862 (дата обращения: 22.08.2024). — Режим доступа: для авториз. пользователей.

Редактор А.А. Литвинова

---

ЛР № 04779 от 18.05.01. В набор В печать

Объем 0,5 усл.п.л., уч.-изд.л. Офсет. Формат 60x84/16.

Бумага тип №3. Заказ № Тираж 75. Цена

Издательский центр ДГТУ
Адрес университета и полиграфического предприятия:
344010, г. Ростов-на-Дону, пл. Гагарина, 1.