**Министерство образования и науки Российской Федерации**

**федеральное государственное бюджетное образовательное учреждение высшего профессионального образования**

**«Российский экономический университет имени Г.В. Плеханова»**

Институт цифровой экономики и информационных технологий

Кафедра информатики

**Практическая работа**

по дисциплине

«Объектно-ориентированные технологии в программной инженерии»

Выполнили

студенты 3 курса

группы 15.27Д-БИ19/22Б

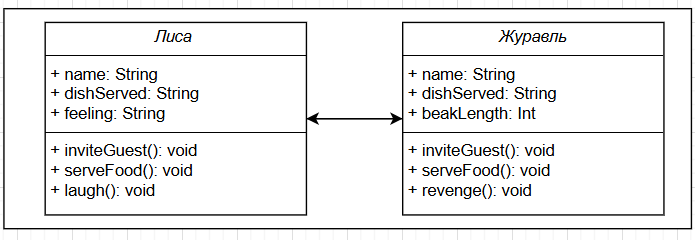
очной формы обучения

высшая школа ВШКМиС

Фамилии студентов Нгуен Као Бач,

Москва, 2025

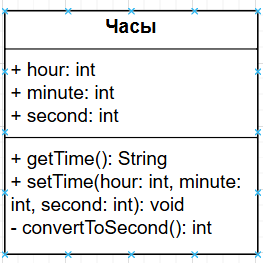
1. Сказка «Лиса и Журавль»



В данной диаграмме классов представлены два основных персонажа сказки «Лиса и Журавль». Класс **«Лиса»** содержит такие атрибуты, как имя, поданное блюдо и настроение, а также методы: пригласить гостя, подать еду и смеяться. Класс **«Журавль»** включает имя, поданное блюдо и длину клюва, а также методы: пригласить гостя, подать еду и отомстить. Между классами существует ассоциация, отражающая их взаимодействие — они приглашают друг друга в гости. Эта диаграмма демонстрирует основные характеристики и поведение персонажей в контексте сказки.

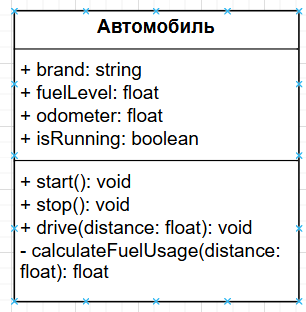
**Задание 2.** Для следующих задач создайте с помощью программы Microsoft Visio диаграмму класса. Класс должен содержать минимум три атрибута, минимум три операции. Укажите типы данных атрибутов, для операций определите уровень инкапсуляции.

**Задача 2.1.** Диаграмма должна описывать класс «Часы». В классе предусмотрены стандартные операции чтения значений атрибутов (геттеры) и установки значений атрибутов (сеттеры). Должна быть предусмотрена возможность изменения времени. Время хранится в часах, минутах и секундах. Включите в класс приватные операции для конвертации времени: пересчет часов, минут и секунд.



Класс **«Часы»** моделирует объект, который хранит и управляет временем в формате часы, минуты и секунды. Он содержит три атрибута: hour, minute и second — все типа int. Класс предоставляет открытые методы getTime() для получения текущего времени в виде строки и setTime() для установки значений времени. Приватный метод convertToSeconds() используется для внутренней логики — он преобразует текущее время в общее количество секунд. Этот метод скрыт от пользователя класса, так как применяется только внутри, например, при сравнении времени.

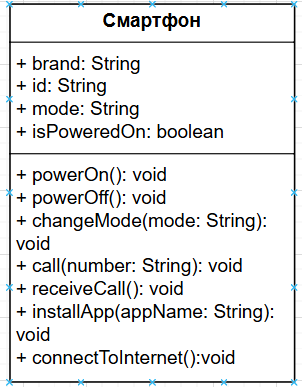
**Задача 2.2.** Диаграмма должна описывать класс «Автомобиль». В автомобиле есть одометр (измеритель пройденного расстояния) и индикатор запаса топлива. Для автомобиля предусмотрите следующие операции: завести, заглушить. Автомобиль при перемещении изменяет свои координаты, сжигает топливо, увеличивает километраж.



Класс **«Автомобиль»** описывает объект машины с основными характеристиками: марка, уровень топлива, пробег и состояние двигателя. Методы позволяют завести и заглушить машину, а также проехать заданное расстояние. При движении изменяется пробег и расходуется топливо. Дополнительно есть приватный метод calculateFuelUsage(), который используется внутри класса для расчёта расхода топлива — он скрыт от внешнего пользователя, что соответствует принципу инкапсуляции.

**Задача 2.3. Диаграмма должна описывать класс «Смартфон».**

Аппарат имеет марку, идентификационный номер, режим работы (без звука, на улице, в самолёте и др.). Смартфон можно включить, выключить и изменить режим вибро на один из предложенных. Также имеется возможность совершить звонок на заданный пользователем номер и ответить на входящий звонок. На смартфон можно устанавливать программное обеспечение и выходить в сеть Интернет.

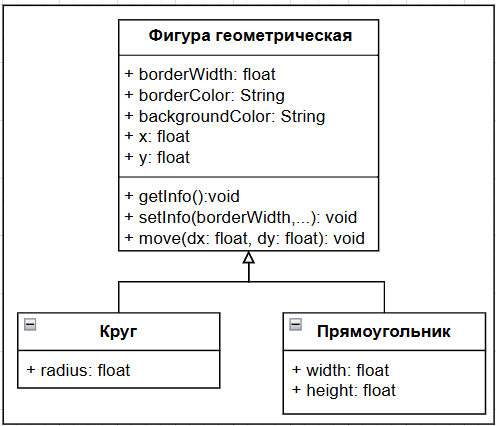


Класс «Смартфон» моделирует современный телефон с основными характеристиками: марка, идентификатор, режим работы и состояние питания. Устройство можно включить, выключить, изменить режим (например, без звука), позвонить, принять вызов, установить приложение и выйти в интернет. Все действия реализованы в виде открытых методов. Такое проектирование отражает функциональность настоящего смартфона.

**Задание 3**. Для следующих задач создайте с помощью программы Microsoft Visio диаграмму минимум трех классов. Второй и третий классы являются наследниками первого, следовательно, наследуют атрибуты и операции от первого. При разработке диаграммы учитывайте ключевые отличия между классами-потомками и общие черты с классом-родителем.

**Задача 3.1.** Диаграмма классов описывает класс-родитель **«Фигура геометрическая»**. Любая фигура должна обладать толщиной линии контура, цветом линии контура, цветом фона, координатами верхней левой точки. Любая фигура обладает операциями, позволяющими считать перечисленные значения и записать их. Геометрическую фигуру можно перемещать, изменяя при этом координаты ее верхней левой точки.

* Первый класс-наследник: **Круг** — обладает радиусом.
* Второй класс-наследник: **Прямоугольник** — обладает высотой и шириной.



Базовый класс **«Фигура геометрическая»** содержит общие атрибуты: borderWidth, borderColor, backgroundColor, а также координаты x и y. Класс предоставляет методы getInfo(), setInfo(borderWidth,...) и move(dx, dy) для получения информации, установки свойств и перемещения фигуры. От него наследуются два подкласса: **«Круг»**, который имеет атрибут radius, и **«Прямоугольник»** с атрибутами width и height. Оба подкласса расширяют функциональность базового класса своими уникальными свойствами.

**Задача 3.2** Диаграмма классов описывает класс-родитель «Машина». Любая машина имеет государственный номер, марку, цвет кузова. Машина способна перемещаться, расходуя при этом топливо.

* Первый класс-наследник: Легковая машина — обладает формой кузова.
* Второй класс-наследник: Грузовая машина — обладает грузоподъёмностью и возможностью перевозить грузы.

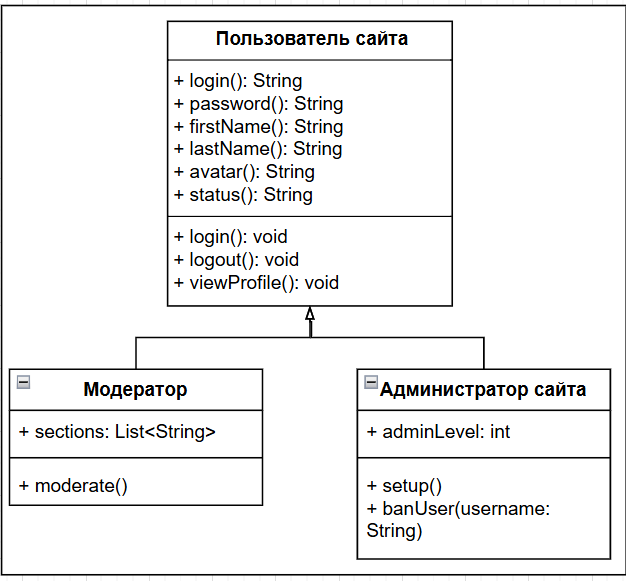
A screenshot of a computer

AI-generated content may be incorrect.

В задаче 3.2 показана иерархия автомобилей. Базовый класс **«Машина»** содержит общие свойства: номер, марка, цвет, а также методы drive() и consumeFuel().  
Класс **«Легковая машина»** добавляет поле bodyType — тип кузова.  
Класс **«Грузовая машина»** имеет maxLoad (грузоподъёмность) и cargoEnabled (может ли перевозить груз).

**Задача 3.3.** Диаграмма классов описывает класс-родитель **«Пользователь сайта»**. Любой пользователь имеет логин, пароль, имя, фамилию и аватар. Кроме того, любой пользователь обладает статусом. Статус по умолчанию — «пользователь».

* Первый класс-наследник: **Модератор** — модерирует созданные им разделы чата, статус «модератор».
* Второй класс-наследник: **Администратор сайта** — выполняет функции по администрированию и настройке работы сайта, статус «администратор».



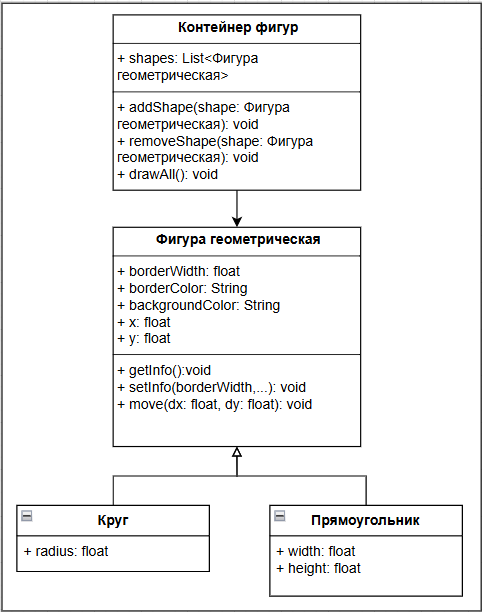
Класс **«Пользователь сайта»** содержит общие данные пользователя: логин, пароль, имя, фамилию, аватар и статус. У него есть методы входа, выхода и просмотра профиля.

Класс **«Модератор»** наследует эти свойства и добавляет список разделов sections, которые он модерирует.

Класс **«Администратор сайта»** также наследует базу, но дополнительно имеет уровень level и методы настройки сайта и блокировки пользователей.

**Задание 4**. Выполните предыдущее задание, указав все связи между классами.  
С помощью каких внутренних атрибутов будут связаны классы?  
Внутри каждого класса выделите атрибут, тип которого соответствует атрибуту в классе, с которым он связан.

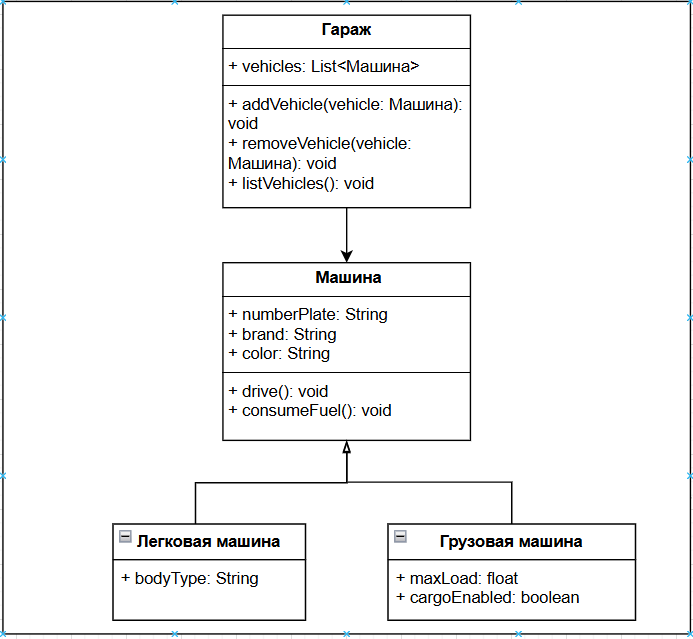
3.1



В задаче 3.1 базовый класс **«Фигура геометрическая»** содержит общие свойства всех фигур: borderWidth, borderColor, backgroundColor, а также координаты x и y. Классы-наследники — **«Круг»** с атрибутом radius и **«Прямоугольник»** с атрибутами width и height — расширяют базовый класс своими характеристиками.

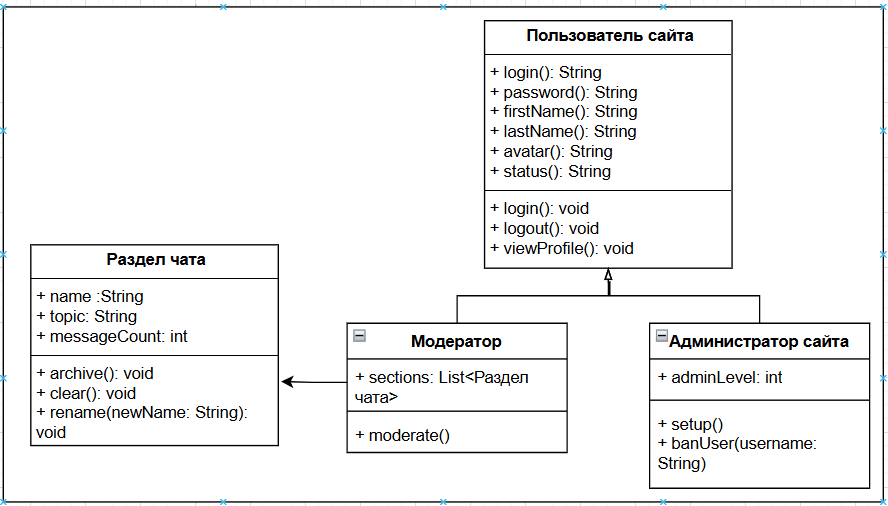
Чтобы отразить связи между объектами, добавлен новый класс **«Контейнер фигур»**, который содержит список фигур через атрибут shapes, тип которого — List<Фигура геометрическая>. Таким образом, реализована ассоциация между контейнером и геометрическими фигурами, что соответствует требованиям задания 4.

3.2



В задаче 3.2 был добавлен класс **«Гараж»**, содержащий список машин через атрибут vehicles типа List<Машина>. Это отражает связь между объектами: один гараж может управлять несколькими машинами. Классы Легковая машина и Грузовая машина наследуют от Машина и расширяют его своими атрибутами.

3.3



Для выполнения задания 4 в модель добавлен класс **«Раздел чата»**, так как модератор управляет несколькими такими разделами. Атрибут sections в классе Модератор теперь представляет собой список объектов Раздел чата.

**Задание 5.**

Создайте диаграмму классов в Microsoft Visio для ниже описанных систем. Выделите сущности. Для каждой сущности разработайте соответствующие классы (минимум пять). Для каждого класса представьте атрибуты (если такое возможно), стандартные операторы «геттеры» и «сеттеры», минимум два специфичных метода. Установите связи между классами. Обоснуйте получившуюся диаграмму.

**Задача 5.1.**

Система «Университет». В университете обучаются студенты. Они могут изучать точные и гуманитарные науки. Преподаватели готовят и проводят занятия лекционные и практические и ведут научную деятельность. Преподаватели работают на кафедрах. Деканат отчисляет неуспевающих.

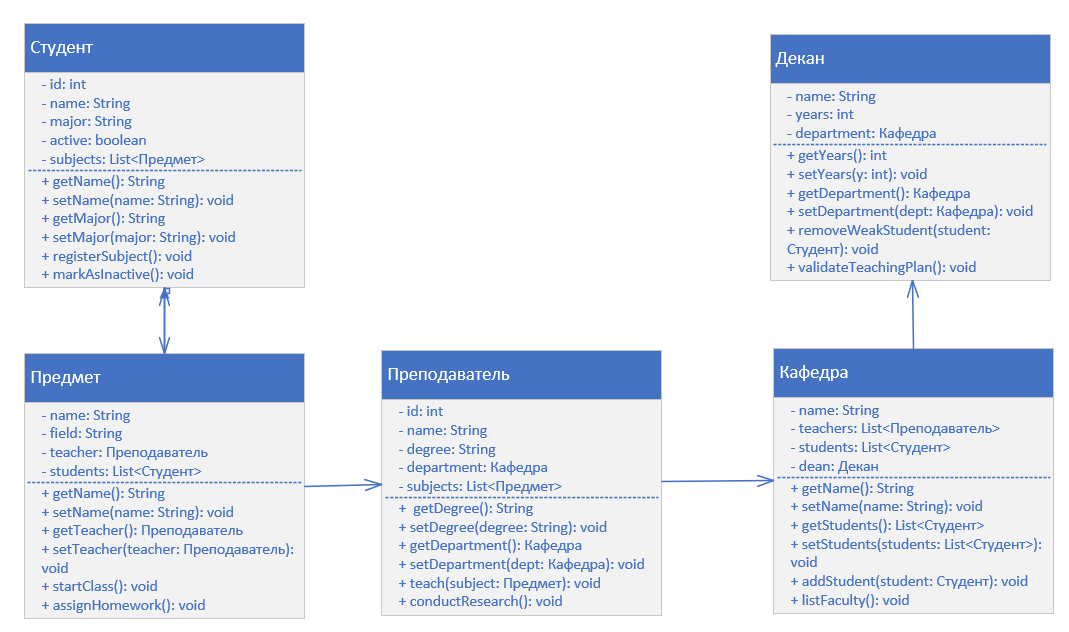


Диаграмма отражает систему «Университет» и включает классы: Студент, Преподаватель, Кафедра, Декан и Предмет. Каждый студент может записываться на несколько предметов, и каждый предмет включает множество студентов, что представляет связь многие-ко-многим. Преподаватель принадлежит к кафедре и ведёт предметы. Каждый предмет ведётся одним преподавателем. Декан возглавляет кафедру, может отчислять слабых студентов и утверждать учебный план. Все классы содержат основные атрибуты, характерные методы, а также стандартные геттеры и сеттеры, что соответствует принципам объектно-ориентированного моделирования.

**Задача 5.2.**

Система «Заправочная станция». Станция предоставляет возможность заправки газом и бензином. Автомобиль подъезжает к колонке с определённым номером. Каждая колонка имеет несколько марок бензина. Водитель оплачивает бензин или газ в кассе. После заправки автомобиль уезжает, тем самым освобождая колонку.

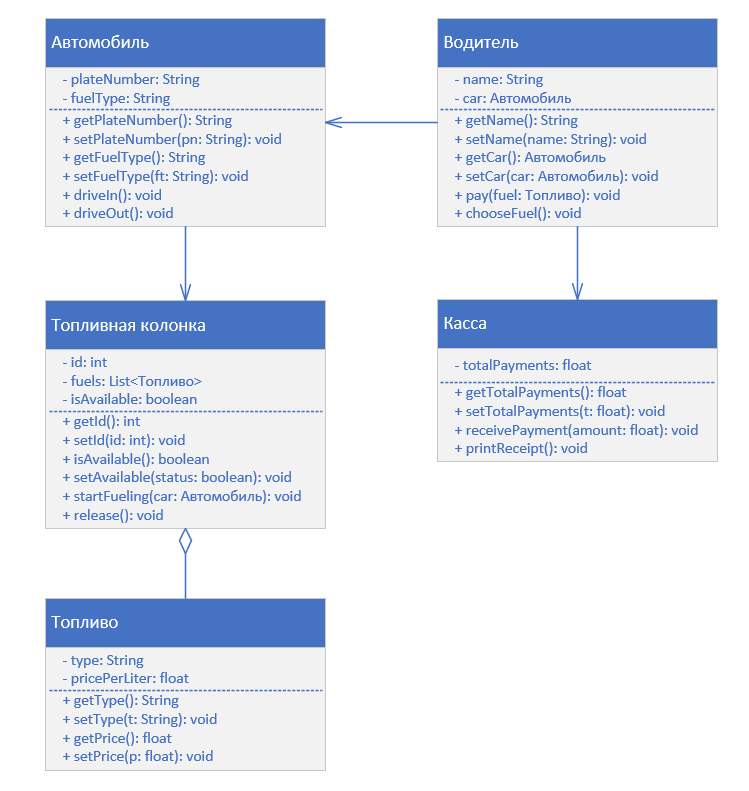


Диаграмма описывает систему автозаправки с основными классами: Водитель, Автомобиль, Топливная колонка, Топливо и Касса. Каждый водитель управляет автомобилем, который подъезжает к конкретной колонке для заправки, а затем оплачивает на кассе. Каждая колонка может предоставлять несколько видов топлива. Эта связь отображается с помощью агрегации — то есть колонка содержит список топлив, но само топливо может существовать независимо от колонки. Все классы содержат необходимые атрибуты, характерные методы, а также геттеры и сеттеры в соответствии с требованиями задания.

**Задача 5.3.**

Система «Туристическое агентство». Клиент выбирает путешествие через агентство. Сначала определяет страну, затем компанию, потом выбирает подходящую путевку. При необходимости оформляется страховка. Клиент оплачивает тур и только после этого получает билеты. Они могут быть бумажными и электронными.

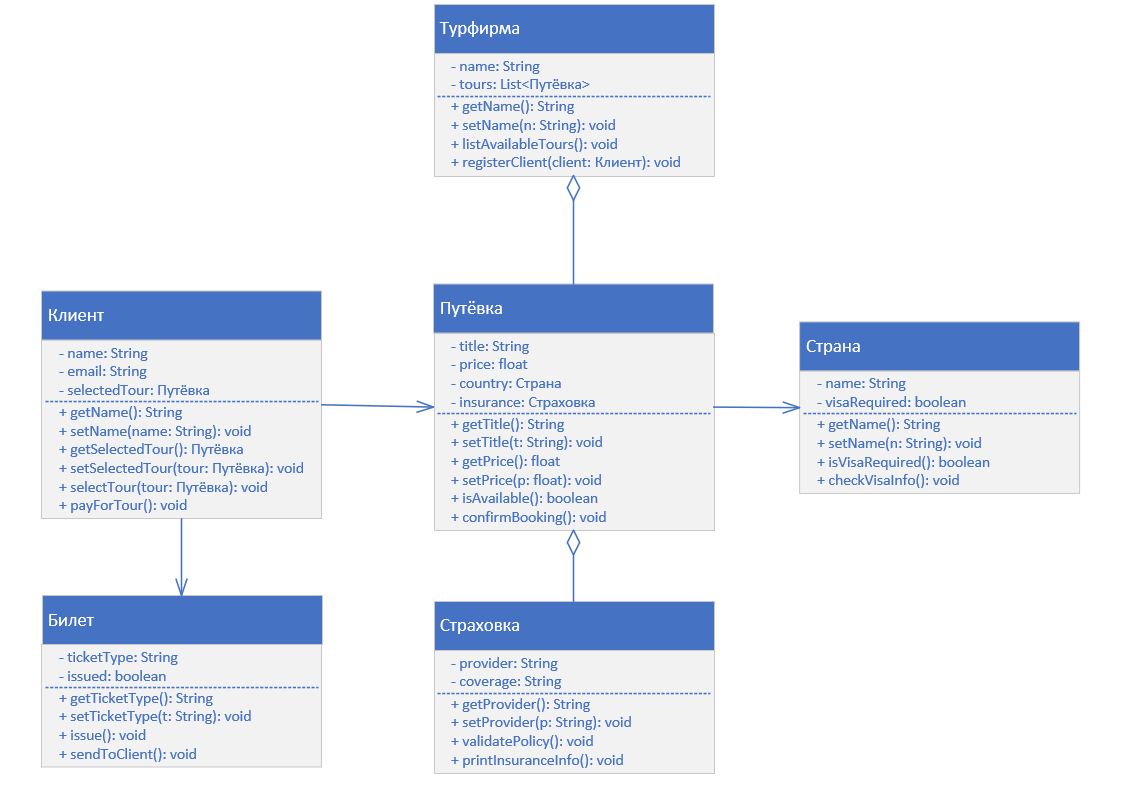


Диаграмма отображает систему туристического агентства с основными классами: Клиент, Путёвка, Турфирма, Страна, Страховка и Билет.  
Клиент выбирает путёвку → это **ассоциация один-к-одному**, обозначенная **прямой стрелкой** от клиента к путёвке.

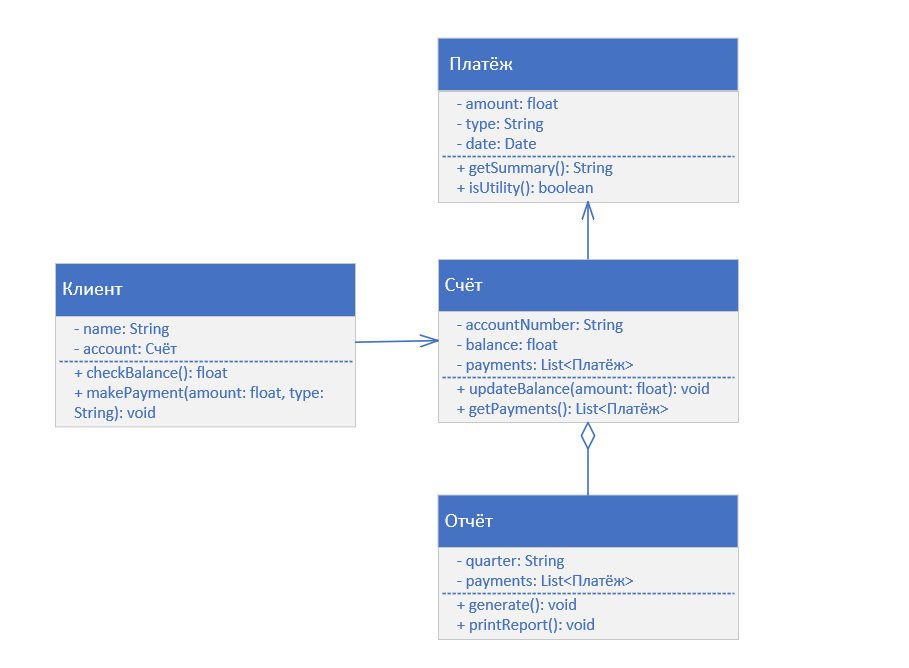
Каждая путёвка привязана к одной стране → **ассоциация (1–1)**.  
Путёвка может содержать страховку, но это необязательное условие → используется **агрегация** (обозначена **белым ромбом** от Путёвка к Страховка).  
После оплаты клиент получает билет → **ассоциация один-к-одному**.  
Одна туристическая компания (Турфирма) предлагает несколько путёвок → **агрегация один-ко-многим** от Турфирма к Путёвка.

**Задание 6.**

Создайте упрощенную диаграмму классов в Microsoft Visio для перечисленных ниже задач. Можно указывать не все атрибуты и операции. Установите связи между классами.

**Задача 6.1.**

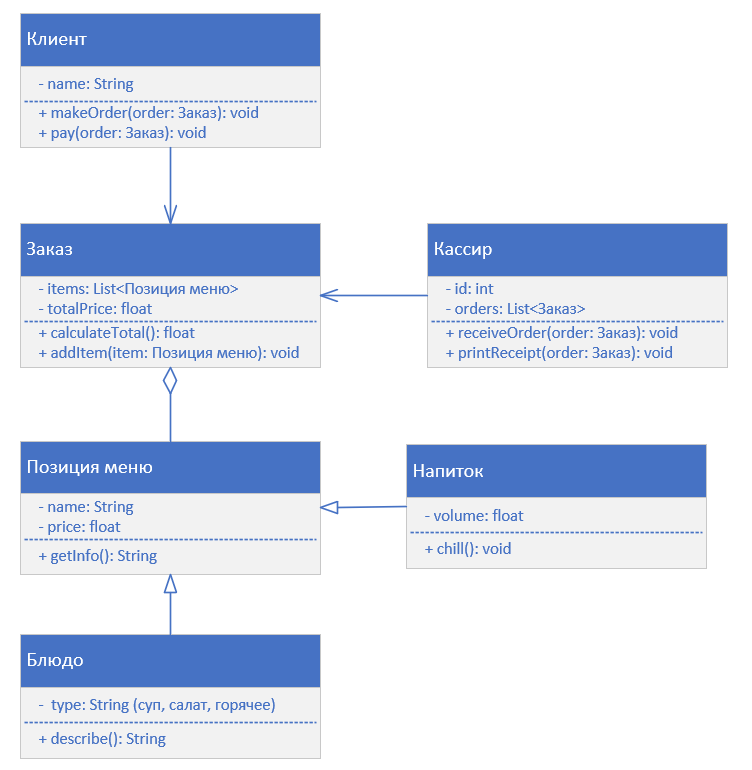
Система «Банк». Клиенты банка имеют там счет. Клиент имеет возможность узнать состояние счета и изменить его, осуществив платеж. Платеж бывает валютный, рублевый или коммунальный. Ежеквартальный отчет состоит из всех платежей за квартал.



Каждый клиент связан с одним счётом (ассоциация 1–1). Счёт содержит список платежей, что отражено с помощью агрегации (1–n), поскольку один счёт может включать несколько платёжных операций. Класс Платёж хранит данные о сумме, типе и дате транзакции. Квартальный отчёт (Отчёт) создаётся на основе списка платежей счёта, поэтому между ними также установлена агрегационная связь.

**Задача 6.2.**

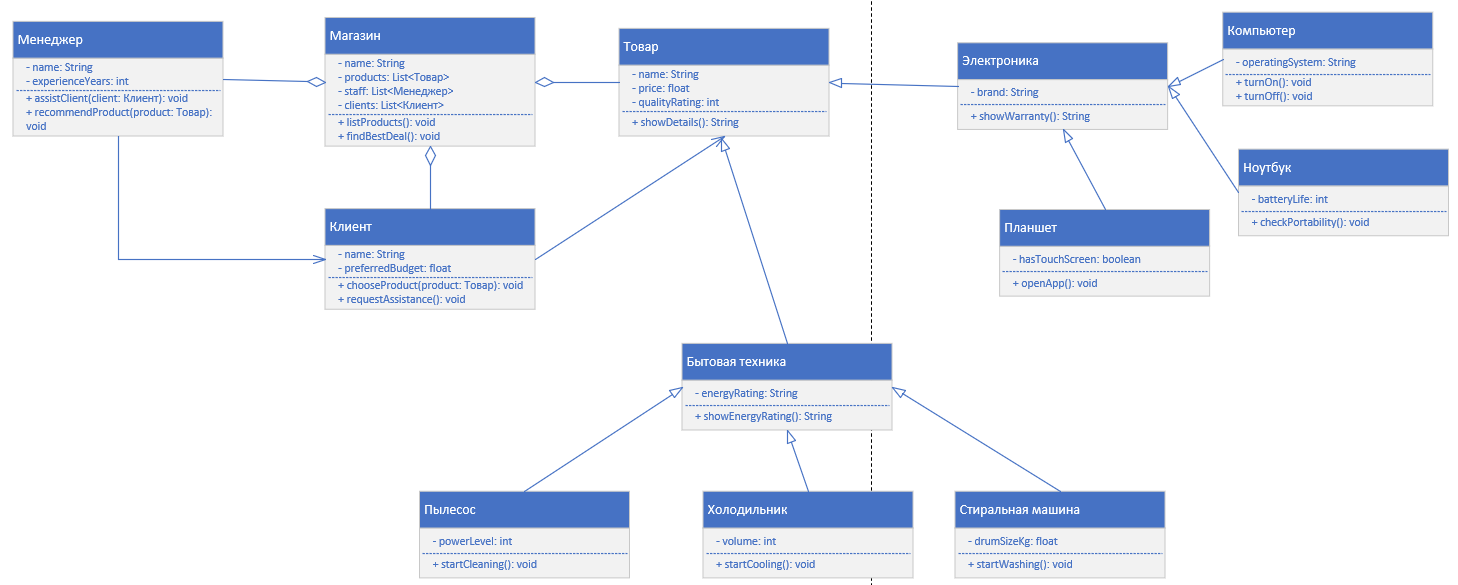
Система «Ресторан быстрого питания». Клиенты при посещении ресторана оплачивают заказы в одной из трех касс. Заказ может состоять из различных блюд и напитков: суп (борщ, солянка, рассольник), салат (оливье, винегрет) и горячее (гарнир и мясо). Напитки представлены в виде чая, кофе, морса и компота.



Класс Клиент представляет клиента, который может создать заказ (Заказ). Каждый заказ включает несколько позиций меню (Позиция меню) через связь **агрегации** (белый ромб), так как позиции могут существовать отдельно. Позиции меню подразделяются на два подкласса — Блюдо и Напиток, которые **наследуются** от Позиция меню.

**Задача 6.3.**

Система «Магазин». Клиенты, посещая магазин электроники, консультируются с менеджерами по продажам и делают для себя лучшее приобретение по цене и качеству. Ассортимент магазина включает в себя электронику и бытовую технику. Электроника включает в себя компьютеры, ноутбуки и планшеты. Бытовая техника: пылесосы, холодильники и стиральные машины.



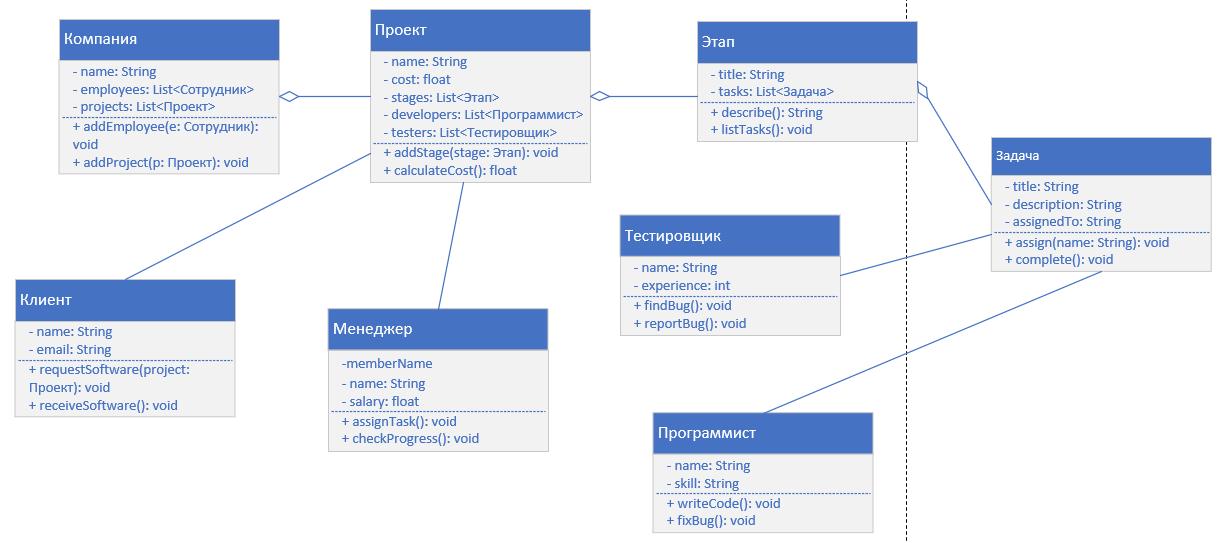
Класс Магазин управляет списками товаров, клиентов и персонала. Товары (Товар) делятся на Электроника и Бытовая техника, каждая из которых включает подкатегории: Компьютер, Ноутбук, Планшет, Пылесос, Холодильник, Стиральная машина. Связи между классами представлены через наследование (пустой треугольник) и агрегацию (белый ромб). Клиенты могут выбрать товар и обратиться за помощью к менеджеру.

**Задание 7.**

Создайте диаграмму классов в Microsoft Visio для перечисленных ниже систем.

**Задача 7.1.**

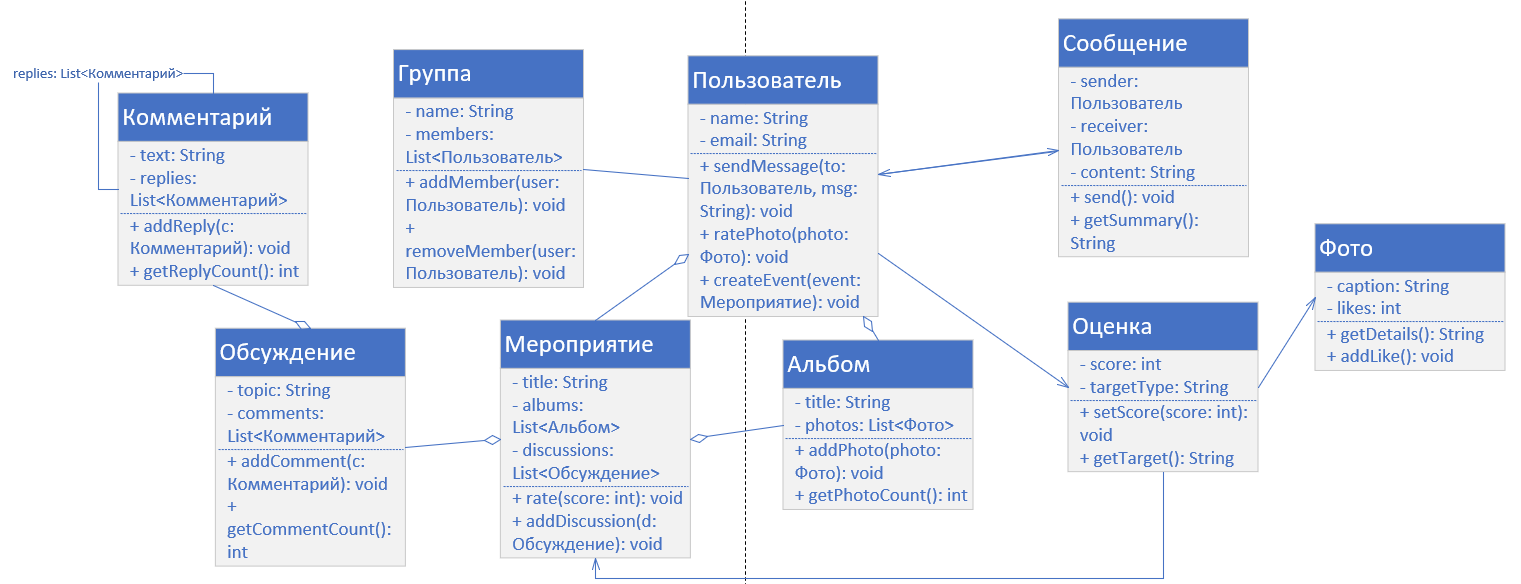
Система «Компания». Компания занимается разработкой программного обеспечения. Сотрудниками компании являются: менеджеры, программисты и тестировщики программного обеспечения. Компания выполняет проекты в определённой последовательности. Перед каждым этапом составляются сценарии использования, описывающий функционал, выполняемый на данном этапе. На основе этих сценариев создаются задачи для выполнения программистами и тестировщиками. Менеджер руководит проектом. Каждый проект несет в себе финансовые траты, включая зарплату сотрудникам. Тестировщик находит ошибки и фиксирует их в отдельном документе. Документация ошибок фиксируется на каждом этапе проекта.



Клиент отправляет запрос в Компанию, которая создаёт Проекты. Каждый Проект состоит из нескольких Этапов, а этапы — из Задач. Менеджер связан с Проектом и отвечает за распределение Задач. Программисты и тестировщики напрямую связаны с задачами, которые они выполняют. Связи между классами отражены с помощью стрелок: агрегация — для составных частей, ассоциация — для взаимодействия.

**Задача 7.2.**

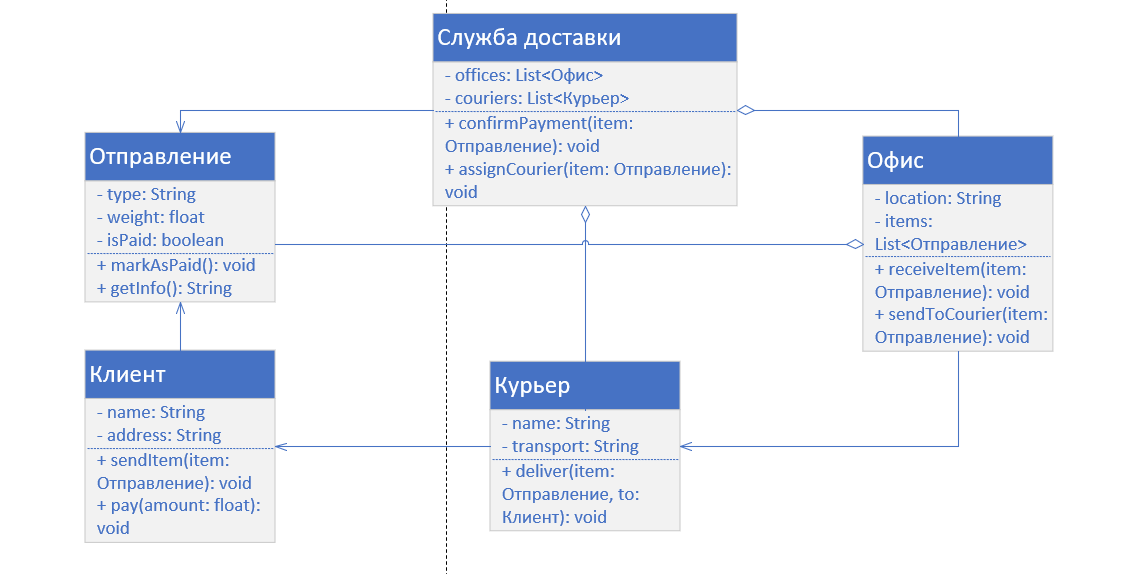
Система «Социальная сеть». Социальная сеть предназначена для организации различных мероприятий. Каждый пользователь сети может хранить альбомы фотографий, состоять в группах, отправлять сообщения другим пользователям, ставить оценки их фотографиям, оставлять комментарии в обсуждениях. Любой пользователь может создавать свои мероприятия. Для мероприятия можно создавать альбомы и обсуждения. В обсуждениях комментарии строятся древовидно. Мероприятие также можно оценить.



Пользователи состоят в группах, комментарии строятся в виде дерева (Комментарий связан с самим собой), мероприятия содержат альбомы и обсуждения. Оценки (Оценка) присваиваются фотографиям и мероприятиям через прямые связи. Связи между классами представлены стрелками: **агрегация** используется для составных частей, а **ассоциация** — для взаимодействий между объектами.

**Задача 7.3.**

Система «Служба доставки». Клиент службы доставки может отправить сообщение в виде конверта, открытки или посылки. Служба принимает отправление после оплаты. Почтовое отправление передаётся в ближайший к адресу офис службы (поездом, самолётом, автомобилем). Затем курьером доставляется адресату домой.



Клиент (Клиент) отправляет один предмет (Отправление) и оплачивает его. Служба доставки (Служба доставки) управляет несколькими офисами (Офис) и курьерами (Курьер). Каждый предмет поступает в ближайший офис, который назначает курьера для доставки по адресу клиента. Связи между классами отображаются через ассоциации и агрегации.

**ЗАДАНИЕ 8.** Индивидуальное. В Microsoft Visio создать диаграмму классов для информационной системы согласно варианту (Приложение). Работа проходит в группе по три человека.

Последовательность работы:

– обсудить необходимые и достаточные списки классов для информационной системы;

– индивидуально в зависимости от варианта разработать отдельные классы (с указанием атрибутов, операций, уровня инкапсуляции);

– в группе соединить все полученные классы в единую схему с указанием различных видов связей (ассоциации, обобщения, агрегации и композиции).

A screenshot of a computer

AI-generated content may be incorrect.

Класс Система управляет пользователями (Пользователь) и группами (Группа). Один пользователь может входить в несколько групп, и каждая группа может содержать нескольких пользователей. Группам назначаются Права, определяющие действия, разрешённые для определённого Ресурса. Связь между Группа и Право представлена как агрегация, что позволяет переиспользовать права. Связь между Право и Ресурс — однонаправленная, показывающая, что право применяется к конкретному ресурсу.

**1. Перечислите особенности диаграммы классов.**

Диаграмма классов UML представляет собой один из видов структурных диаграмм. Она описывает статическую структуру системы, включая классы, их атрибуты, методы и взаимосвязи между ними. Диаграмма классов используется для анализа предметной области, проектирования архитектуры программного обеспечения и может служить основой для автогенерации кода.

**2. С какой целью создаются диаграммы классов?**

Диаграммы классов создаются для описания структуры системы в терминах классов и их связей. Основные цели:

* моделирование предметной области,
* проектирование программного обеспечения,
* визуализация компонентов системы,
* обеспечение общего понимания между разработчиками,
* генерация кода на основе модели.

**3. Перечислите элементы нотации диаграммы классов.**

Основные элементы нотации диаграммы классов:

* **Класс** — прямоугольник, разделённый на три части: имя, атрибуты, методы.
* **Атрибуты** — свойства объекта, записываются как + имя: тип.
* **Методы** — действия или поведение, записываются как + метод(параметры): тип.
* **Связи между классами** — наследование, ассоциация, агрегация, композиция и зависимость.
* **Модификаторы доступа**: + public, - private, # protected.

**4. Назовите уровни инкапсуляции для атрибутов и методов.**

Существуют три основных уровня инкапсуляции:

* + (public): открытый доступ — виден всем.
* - (private): доступ только внутри класса.
* # (protected): доступен классу и его подклассам.

Они используются для контроля доступа к данным и методам объекта, реализуя принцип инкапсуляции в ООП.

**5. Какие виды связей существуют между классами? Приведите примеры.**

В UML выделяют следующие виды связей:

* **Обобщение (наследование)**: подкласс наследует свойства и методы суперкласса (например, Птица ← Животное).
* **Ассоциация**: общая связь между объектами, может быть одно- или двунаправленной (Пользователь — Альбом).
* **Агрегация**: "целое–часть", где части могут существовать независимо (Компания содержит Сотрудников).
* **Композиция**: сильная форма агрегации, где части не могут существовать без целого (Окно — Кнопка).
* **Зависимость**: один класс использует функциональность другого (Медиаплеер зависит от Кодека).