|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

**КУРСОВАЯ РАБОТА**

по дисциплине «Объектно-ориентированное программирование»

**Тема курсовой работы** Проектирование и реализация программной системы с использованием объектного подхода

**Студент группы** ИКБО-07-17 Акжигитов Радмир Русланович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

**Руководитель курсовой работы** ст. преподаватель Гусев К.В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Работа представлена к защите «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2018 г.

Допущен к защите «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2018 г.

Москва 2018

|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

Утверждаю

Заведующий кафедрой МОСИТ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Головин С.А.

«\_\_\_» \_\_\_\_\_\_\_\_\_ 2018 г.

**ЗАДАНИЕ**

**на выполнение курсовой работы по дисциплине**

«Объектно-ориентированное программирование»

Студент Акжигитов Радмир Русланович Группа ИКБО-07-17

**Тема работы:** Проектирование и реализация программной системы с использованием объектного подхода

**Исходные данные:** задание на курсовую работу,тема №1, вариант № 10. Разработать систему Личный кабинет квартиросъемщика. Служба госуслуг каждый месяц выставляет счет по оплате коммунальных услуг: за расход горячей и холодной воды, за эксплуатацию помещения (площади), взнос на капитальный ремонт. Объем расхода горячей и холодной воды рассчитываются на основе показателей счетчиков учета расхода воды, которые квартиросъемщик обязан отправить до 20 числа текущего месяца в единый расчетный центр госуслуг.

**Функционал:** регистрация квартиросъемщика, с указанием логина, пароля, Фамилии, Имени, Отчества, полного адреса, общую площадь квартиры, количество проживающих в квартире, вид льгот по оплате коммунальных слуг квартиросемщика; вход в созданный личный кабинет по логину и паролю; ввод показаний счетчиков воды за текущий месяц; формирование службой единого центра счета к плате за текущий месяц; отображение всех плат за текущий год; сохранение данных истекшего года в файле с именем Номер года; ведение нормативов на оплату каждого вида услуг.

**Перечень вопросов, подлежащих разработке, и обязательного графического материала:**

Моделирование объектной системы с использованием обозначений языка моделирования UML

Реализация программной системы средствами языка С++

Отчет по курсовой работе в виде расчетно-пояснительной записки .

**Срок представления к защите курсовой работы:** до «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2018 г.

**Задание на курсовую работу выдал** \_\_\_\_\_\_\_\_\_\_\_\_ (Гусев К.В.)

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2018 г.

**Задание на курсовую работу получил**  \_\_\_\_\_\_\_\_\_\_\_\_(Акжигитов Р. Р.)

[РЕФЕРАТ 5](#_Toc515417348)

[ВВЕДЕНИЕ 6](#_Toc515417349)

[ТЕОРЕТИЧЕСКОЕ ВВЕДЕНИЕ 7](#_Toc515417350)

[Объектно-ориентированное программирование 7](#_Toc515417351)

[Основные понятия 7](#_Toc515417352)

[Описание UML 9](#_Toc515417353)

[Что такое UML 9](#_Toc515417354)

[Структура диаграммы класса 9](#_Toc515417355)

[Атрибуты, методы и обозначение доступа 10](#_Toc515417356)

[Наследование и интерфейсы 10](#_Toc515417357)

[Композиция: агрегация и ассоциация 11](#_Toc515417358)

[Кардинальность 13](#_Toc515417359)

[Метод создания интерфейсов 13](#_Toc515417360)

[JSON 14](#_Toc515417361)

[ПРОЕКТНАЯ ЧАСТЬ 16](#_Toc515417362)

[Постановка задачи 16](#_Toc515417363)

РЕФЕРАТ

ВВЕДЕНИЕ

Тема, выбранная мной, актуальна, так как ее реализация позволит соединить центры гос. услуг и квартирантов (квартиросъемщиков). Людей, состоящих в этих отношениях, очень много, ведь все жильцы платят коммунальные услуги, оплачивают ремонт. Автоматизация этого процесса – востребованная и тяжелая задача.

Цель данной курсовой работы состоит в разработке программного продукта, способного решить или упростить задачу сбора данных с собственников квартир, вести временную статистику, отправленных платежей, дать возможность вносить правки в услуги (название, тариф и т.д.).

Реализация программной части на языке C++ в среде разработки Microsoft Visual Studio 2017.

В ходе курсовой работы были спроектированы модули и их функции, отвечающие за каждую часть программы. Было спроектировано взаимодействие классов между собой.

Данная курсовая работа преследует за собой цели:

1. Обучение элементам технологии создания программного продукта и применение их на практике;
2. Использование объектно-ориентированного подхода в ходе решения прикладной задачи;
3. Оформление документации для программного обеспечения.

Данная работа демонстрирует создание и работу с собственными программными объектами на примере взаимоотношений собственников и портала гос. услуг.

ТЕОРЕТИЧЕСКОЕ ВВЕДЕНИЕ

# Объектно-ориентированное программирование

Объектно-ориентированное программирование – один из подходов программирования, который включает в себя такие понятия как: класс, объект, взаимоотношения между классами и объектами, основной принцип: разбиение сложного механизма (всего алгоритма программы) на логические составные части (классы) и наделение объектов класса методами, присущими только им.

Основные понятия

Объект (экземпляр класса) - сущность, которая обладает набором свойств, определенными в классе и поведением, описанным в классе. Объектно-ориентированная программа включает себя множество взаимодействующих определенным образом объектов.

Класс – трафарет (прототип) объекта, содержащий все методы и атрибуты, которые получат будущие объекты. Класс можно назвать типом данных более высокого уровня, так как он будет в объявлении своем содержать базовые типы и функции взаимодействия.

Инкапсуляция – свойство системы, состоящее в том, что объект скрывает внутренние данные и алгоритмы поведения (детали) от других объектов, предоставляет их через интерфейсы взаимодействия. Надежные классы всегда проектируются с учетом инкапсуляции.

Интерфейс – основные средства коммуникации между объектами. Любой метод, который содержится в объекте, должен вызываться через интерфейс. Объект, содержащий атрибут, управляет доступом к нему. Это вопрос безопасности данных. Если мы предусмотрим установку атрибутов, только в одном месте, с помощью сеттера, то нам не придется отслеживать каждый фрагмент кода, который мог изменить соответствующее поле. Особое значение следует уделить таким моментам, как передача личной информации и паролей.

Наследование – свойство системы, позволяющее описать новый «внутренний» (дочерний) класс на базе существующего «внешнего» класса (базового, родительского). Дочерний класс обладает полями и функциями производного (полностью или частично, используя модификаторы доступа), а также может иметь свои собственные атрибуты и методы. Этот принцип позволяет создавать абсолютно новые классы путем абстрагирования общих атрибутов и поведений. Создав подкласс, мы можем использовать его в качестве суперкласса и создавать классы наследников уже от него.

У класса может иметься только один базовый, но много дочерних – в таком случае, это простое наследование, если родителей класса может быть несколько, то это множественное наследование.

Пример использования наследования классов (Рисунок 1): базовый класс Mammal (млекопитающее) имеет 2 подкласса: Dog (собака) и Cat (кошка), в свою очередь класс Dog может включать в себя подвиды собак: German Shepherd и Poodle. Каждый уровень ниже наследует все параметры (атрибуты и методы) высших уровней.

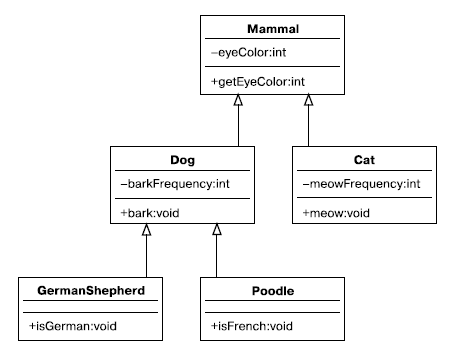


Рисунок 1 - Пример наследования

Полиморфизм – свойство системы, которое позволяет разным объектам по-разному «реагировать» на одно и то же сообщение. Множественное поведение на один и тот же интерфейс у объектов, у которых нам не обязательно знать их внутреннюю структуру и типы данных.

Композиция – возможность одних объектов содержать множество других объектов. Например, компьютер – это сложный объект, он содержит жесткий диск, который тоже является объектом.

# Описание UML

Что такое UML

UML – унифицированный графический язык для моделирования, визуализации и конструирования главных сущностей (абстрактной модели) и связей в архитектурной части программного обеспечения.

Объектное моделирование – является часть. языка UML.

Структура диаграммы класса

Диаграмма класса включает в себя 3 части: имя класса, атрибуты и методы (конструкторы – тоже методы). Класс, отображенный на диаграмме является прямоугольником, который разделен горизонтальными линиями на 3 вышеописанные части. Пример на Рисунок 2 ниже.

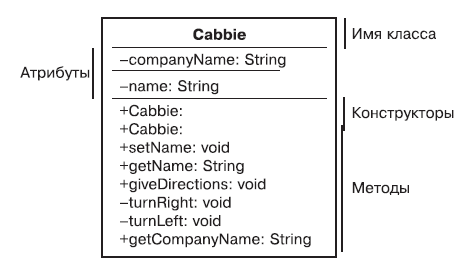


Рисунок 2 – Компоненты диаграммы класса

Атрибуты, методы и обозначение доступа

Для отображения атрибутов и методов на диаграмме классов необходимо написать:

1. Модификатор доступа, в виде «+» - public (поле видно в классе, вне класса, доступно для наследников), «-» - private (к полю можно обращаться только в классе, недоступно для наследников), «#» - protected (поле видно в классе и для наследников).
2. Название поля (атрибута, функции), также можно включить список параметров.
3. Тип поля (тип свойства или возвращаемый тип метода). Для конструкторов тип не указывают.

Пример:

+Cabbie – конструктор

+giveDirections(param1, param2) : void – public метод

-processSecretInformation() : String – private метод, возвращающий String.

Наследование и интерфейсы

Рассмотрим различные виды возможных связей между несколькими объектами. Так как наследование и интерфейсы тесно связаны и их отображение сильно похоже рассмотрим их вместе.

На диаграмме классов наследование представляется пустой сплошной стрелкой, указывающей от дочернего класса к родительскому (Рисунок 3).

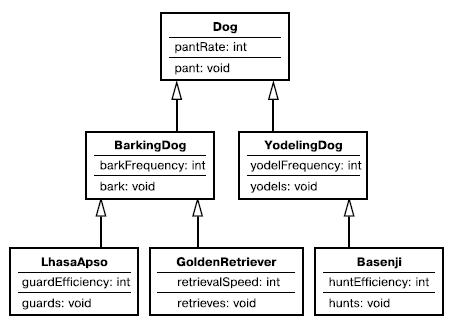


Рисунок 3 – Пример наследования UML

Отношение интерфейс объекта обозначается пустой штриховой стрелкой (Рисунок 4).

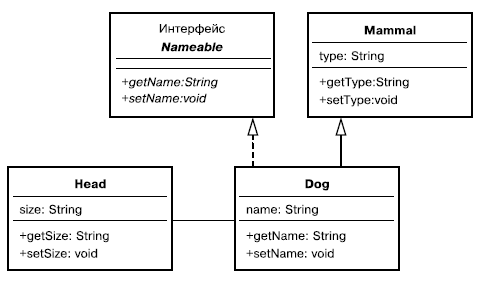


Рисунок 4 – UML-отношение интерфейса

Композиция: агрегация и ассоциация

Композиция – вариант использования отношения «содержит как часть» («является экземпляром» (наследование) не подходит при проектировании).

При композиции класс создается с применением других классов.

Есть два варианта композиции – агрегация и ассоциация. При агрегации класс является частью другого класса (покрышка – автомобиль, покрышка существует только совместно с классом автомобиль (Рисунок 5)). На диаграмме отмечается сплошной линией с пустым «ромбиком» на конце (в направлении совокупляющего класса). Существует строгий вид агрегации – композиция, при которой один объект жестко связан с другим, объект не может существовать без своего класса, к которому он прикреплен. Также ограничен временем жизни внешнем классом. На схеме отображается «закрашенным» ромбиком.

При ассоциации классы взаимодействуют между собой, но могут существовать друг без друга (не являются частью друг друга). Пример, клиент – сервер, существуют отдельно, но обмениваются друг с другом информацией (Рисунок 6). Отображается сплошной линией. Направление отсутствует.

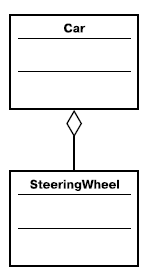


Рисунок 5 – UML-отношение агрегации

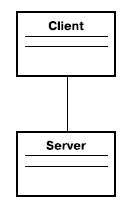


Рисунок 6 – UML-отношение ассоциации

Кардинальность

Кардинальность – диапазон объектов (количество возможных связей), соответствующих определенному классу. Например, в состав компьютера должна входить одна и только одна материнская плата (кардинальность 1 к 1). Однако планок оперативной памяти можно установить несколько, минимум одну (1 компьютер к 1..\* или + хотя бы одной планке памяти).

Предельные значения кардинальности. Если известно, что в данной плане предусмотрено 4 слота для установки памяти, но вместо \* или N, у нас будет число 4 (1 компьютер к 1..4 планок).

Кардинальность может иметь оба «направления» действия. Пример: 1 человек может работать на 0..\* компаний, однако на 1 компанию могут работать 1..\* человек.

# Метод создания интерфейсов

Существует два разных способа создания интерфейсов:

1. Описание графических элементов на языке верстки. Соединение написанного макета с кодом программы. Пример: python-библиотека Kivy App с языком верстки интерфейса YAML. Этот способ менее нагляден, но позволяет гибко настроить пользовательские элементы, имеет множество вариантов комбинирования элементов в контейнерах.
2. Конструирование интерфейса в графическом редакторе. Данный метод позволяет сразу увидеть будущий результат. Пример: winForms на С# и QT C++.

В данной работе я буду использовать WinForms на С++.

Редактор состоит из множества элементов управления, такие как кнопки, текстовые поля, файловый проводник и др. Каждый элемент имеет свои свойства такие как: размер, положение на форме, цвет, текст, отображаемый на элементе.

Также любой элемент имеет свои события: нажатие ЛКМ и ПКМ, наведение курсора на элемент, перетаскивание файла и др. На событие можно установить пользовательский обработчик (функцию), который выполнится при происшествии события. В функции обработчике можно изменять свойства любого элемента управления на расположенного на форме.

Эта среда вполне подходит для данной задачи.

# JSON

JSON – формат, используемый для передачи данных в виде текста, в основе лежит нотация объектов языка JavaScript, не зависит от языка, так как имеет парсеры почти на любой язык, является легким для понимания и простым для прочтения.

Его синтаксис очень прост (Рисунок 7). Объект заключается в фигурные скобки. Содержимое – множество ключей, значения которых могут быть: строкой, числом, логической константой, массивом, другим объектом. Формат поддерживает сколь угодно большую вложенность, что позволяет описывать и передавать объекты любой сложности. Представляет собой ассоциативный массив.

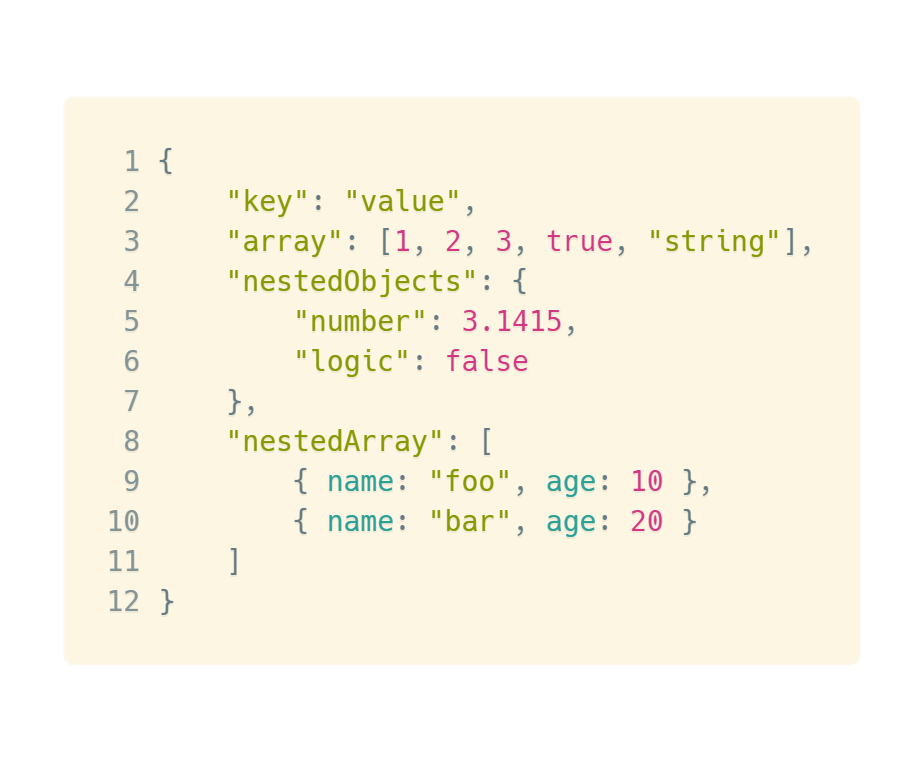


Рисунок 7 – Синтаксис JSON

ПРОЕКТНАЯ ЧАСТЬ

# Постановка задачи

Квартиросъемщик должен платить за услуги каждый месяц (вода, газ, ремонт, аренда). Ему приходит запрос от портала гос. услуг, на который он должен до 20 числа каждого месяца отправить показания счетчиков, если они нужны. Центр гос. услуг в ответ отправляет квитанцию на оплату, посчитанную специальным образом, в зависимости по какому критерию рассчитывается стоимость услуги: по показаниям счетчика, по нормативу, по площади жилья. Квартирант должен оплатить все услуги, только затем он может переключиться на следующий месяц, и цикл повторяется.

Необходимый функционал

1. Квартиросъемщик ОТПРАВЛЯЕТ показания в портал услуг, ОПЛАЧИВАЕТ счета, полученные от портала.
2. Портал услуг ПОЛУЧАЕТ показания собственника, ВЫСЧИТЫВАЕТ стоимости и ОТПРАВЛЯЕТ запрос на оплату обратно.
3. Портал услуг может также ИЗМЕНЯТЬ, ДОБАВЛЯТЬ и УДАЛЯТЬ услуги к оплате.
4. ВЕДЕНИЕ порталом услуг статистики платежей, СОХРАНЕНИЕ устаревших платежей в файл.
5. РЕГИСТРАЦИЯ и АВТОРИЗАЦИЯ в базе данных новых пользователей.