* Модификаторы доступа
  + Public – доступ во всех классах и пакетах
  + Private – доступ только в классе (решение сеттеры/геттеры)
  + Protected – видны для всех классов наследников и всех классов в данном пакете
  + Default – none – всем классам в текущем пакете
* This/super
  + This - ссылается на текущий экземпляр класса, в котором она используется
  + Super – ссылается на текущий экземпляр родительского класса.
* Import com.c2.\*; / Import java\*;импорт только в начале
* Основные принципы ООП
  + Класс — это описание еще не созданного объекта, как бы общий шаблон, состоящий из полей, методов и конструктора
  + Объект – экземпляр класса, созданный на основе этого описания.
  + **Абстракция** В абстрактном классе также можно определить поля и методы, но в то же время нельзя создать объект или экземпляр абстрактного класса. Абстрактные классы призваны предоставлять базовый функционал для классов-наследников. А производные классы уже реализуют этот функционал. С помощью абстракции мы выделяем общее для всех объектов. Abstract
  + **Инкапсуляция** — это сокрытие реализации класса и отделение его внутреннего представления от внешнего. Нельзя полностью закрыть доступ к объекту – он станет бесполезным. Поэтому эту проблему решают сеттеры/геттеры – открытые методы, которые работают со скрытыми полями
  + **Наследование** заключается в использовании существующих классов для описания новых. В Джаве запрещено множественное наследование. Extends
  + **Полиморфизм** - возможность переопределить в классе наследнике унаследованные методы. Полиморфизм Animal animal = new Dog(); (только если Dog наследован от Animal) имеем доступ только к методам от А, но если метод переопределён в B то вызовется именно он.(upcasting – восходящее преобразование так как Animal -> Dog) Dog Dog1 = (Dog) Animal (downcasting – нисходящее преобразование так как Animal -> Dog)