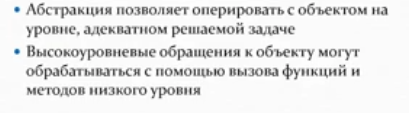
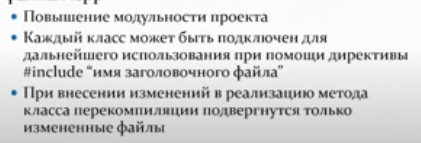
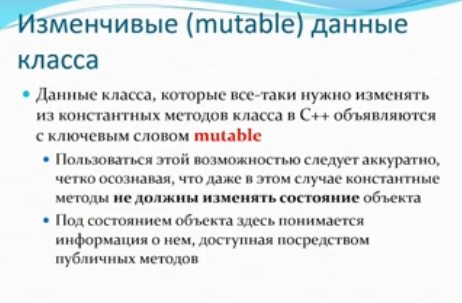
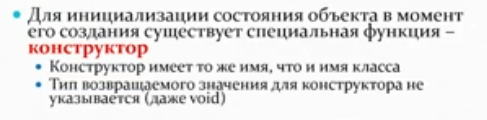
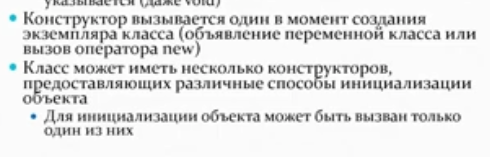
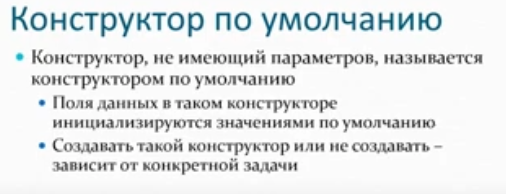
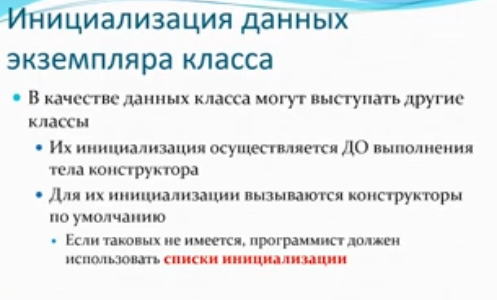
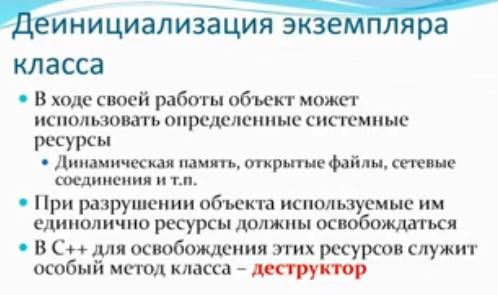
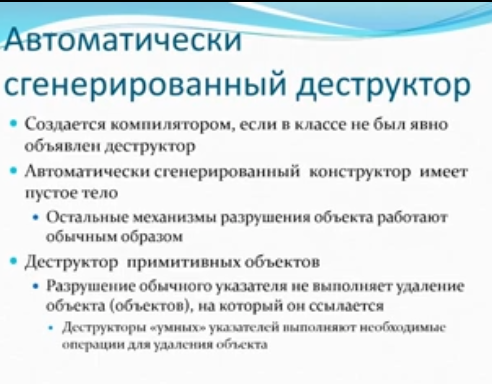
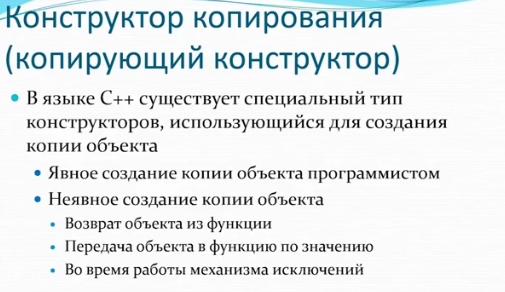
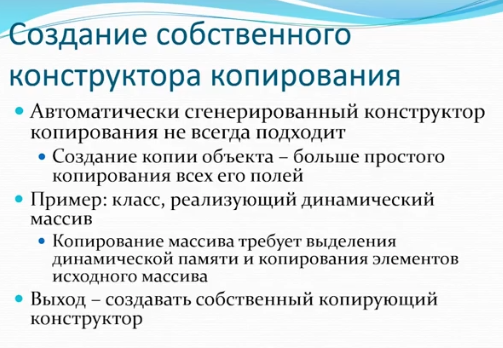
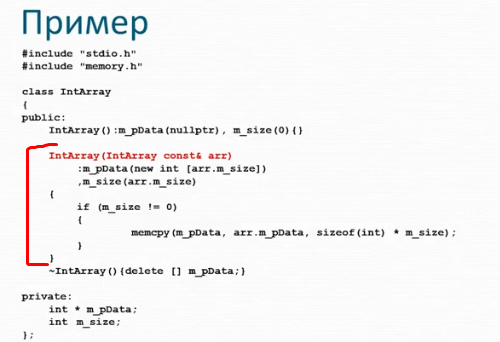
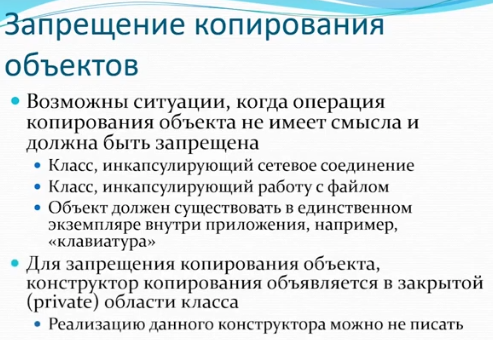
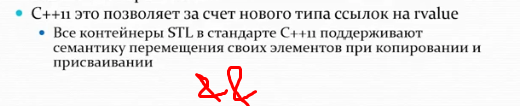
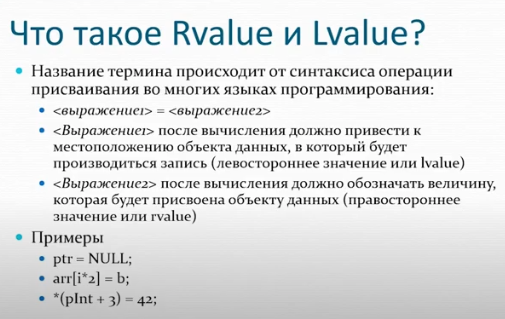
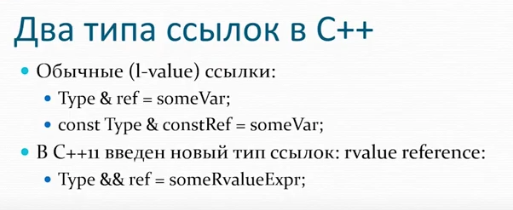
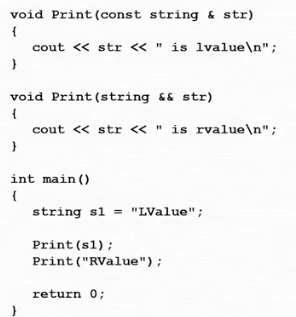
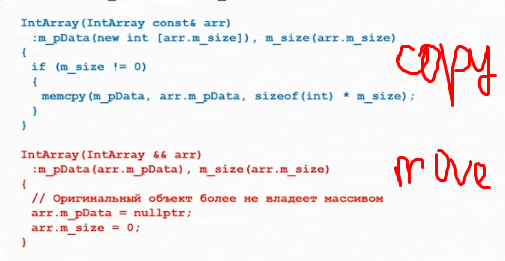
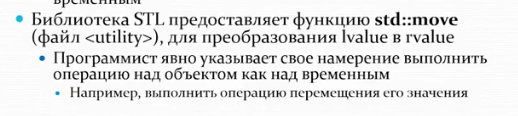
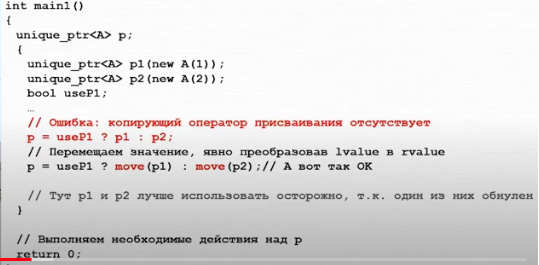
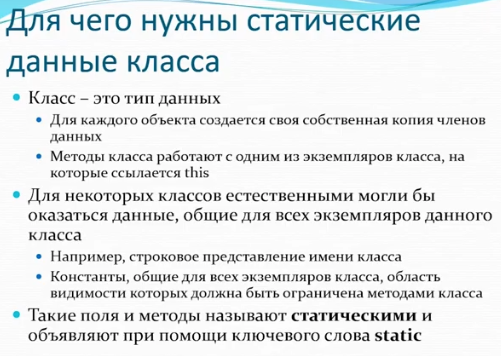
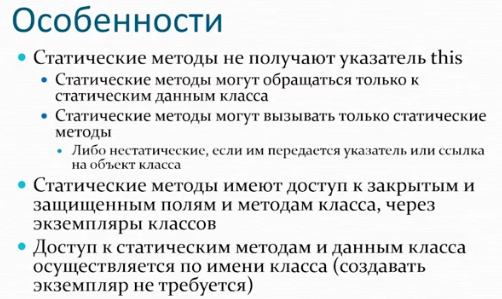
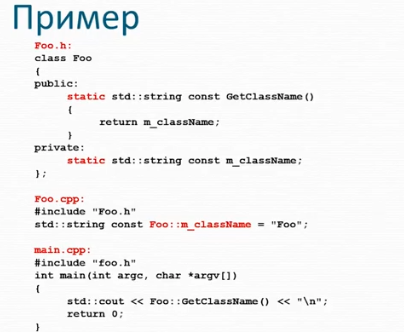
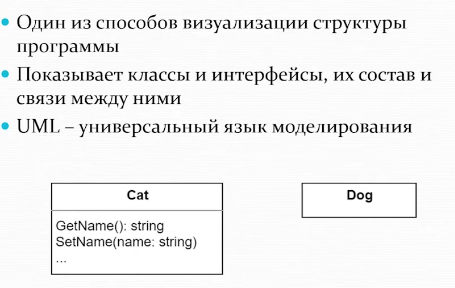
  
  
  
  
**Что такое ООП?**  
Объектно-ориентированное программирование.  
Основано на классах и объектах(инстенс класса).  
  
**Члены-данные** – хранят всю информацию о состоянии объекта (m\_age, m\_name).  
  
**Свойства** – составляющая часть объекта.  
  
**Методы класса** – манипуляция над данными объекта.  
Похож на функцию или процедуру, но контекст выполнения это экземпляр класса (объект).  
  
**Основные принципы ООП:**  
Абстракция, инкапсуляция, наследование, полиморфизм.  
  
**- Абстракция.**  
Заключает в том, что объекты предоставляют неполную информацию о сущностях (для класса студент не важен цвет волос).  
  
  
**- Инкапсуляция.**  
Способность объекта скрывать внутреннее устройство своих свойств и методов.  
Для чего? Чтобы у пользователя был только предусмотренный нами интерфейс для работы (При переключении каналов на пульте, пользователь не задумывается об устройстве пульта).

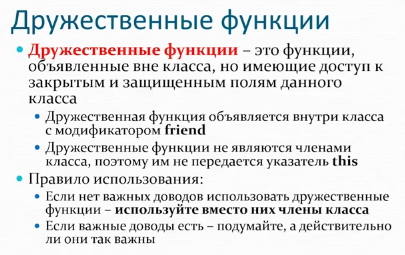
**- Наследование.**  
Позволяет описать новый класс на основе уже существующего родительского (базового) класса.  
Помогает строить иерархии классов.  
Помогает при общих данных в нескольких классах (класс рабочий, студент, т.д. это классы потомки от класса человек).  
  
**- Полиморфизм.**  
Явление, при котором классы-потомки могут изменять реализацию метода класса-предка, сохраняя интерфейс.  
Пример: изменение метода GetArea у классов потомков от класса Shape.  
  
  
**Размещение классов в различных файлах.**  
Общепринятая практика.  
  
Объявление классов в заголовочных файлах .h.  
Реализация в файлах .cpp.  
  
  
  
**Модификаторы доступа.**  
public, private, protected.  
  
**Переопределение методов.**  
Метод виртуальный – может быть переопределён классами наследниками.

Чисто виртуальный метод (= 0), значит, что вообще неопределенна реализация (Нельзя создать экземпляр класса с чисто виртуальным методом).

Если у класса есть чисто виртуальный метод, то этот класс – абстрактный.  
Переопределение происходит с помощью слова override.  
  
  
  
  
  
  
  
  
  
Константные методы – не изменяет состояние объекта. Нельзя в них менять поля класса.  
  
Зачем? Чтобы пользователь знал, что объект не изменится.  
  
В метод класса неявно передаётся указатель на объект, для которого он вызывается. Из-за этого метод применяется именно к нужному объекту.  
Ссылка на себя. This.  
  
  
  
  
  
  
  
  
  
Конструктор.  
  
  
  
Может быть много конструкторов.

  
  
  
  
Список инициализации.  
  
  
Деструктор.  
  
  
В деструкторе освобождать ресурсы.  
  
  
Копирующий конструктор.  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
Перемещающий конструктор.  
  
  
  
  
  
  
  
  
  
Перемещающий конструктор будет принимать значение по RValue ссылке.  
  
  
  
  
  
  
  
  
  
  
  


  
  
**Статические данные и методы класса.**  
Нужны данные для всех объектов класса.  
  
  
  
  
  
  
Статические методы не могут быть const, ведь они не меняют какой-либо объект.  
  
**Отношения между классами.**  
  
- Диаграммы классов.  


**Дружественные функции и классы.**Более тесная связь, чем наследование.  
  
  
  
  
  
Отношение дружбы в одностороннем порядке.  
  
