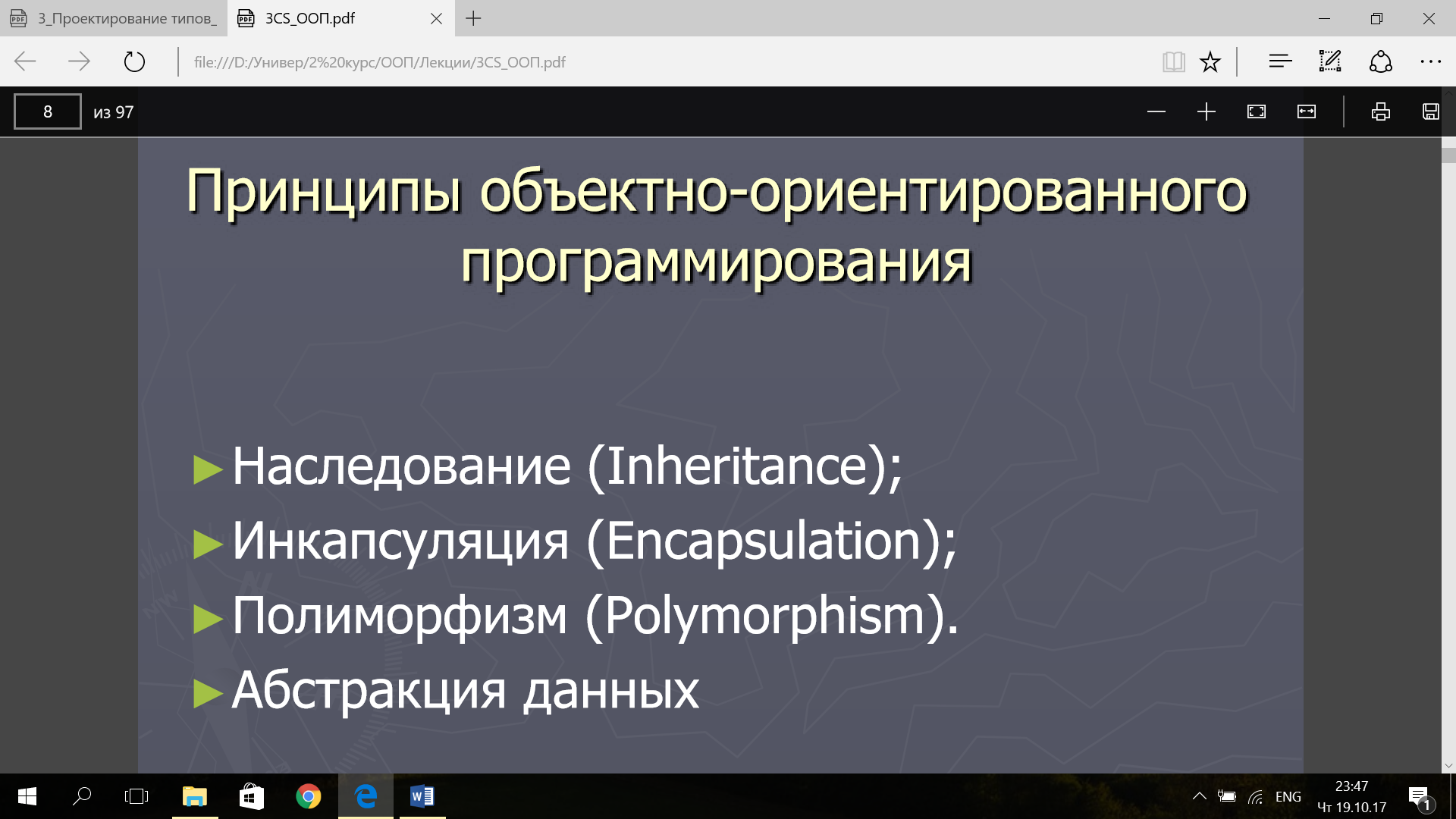
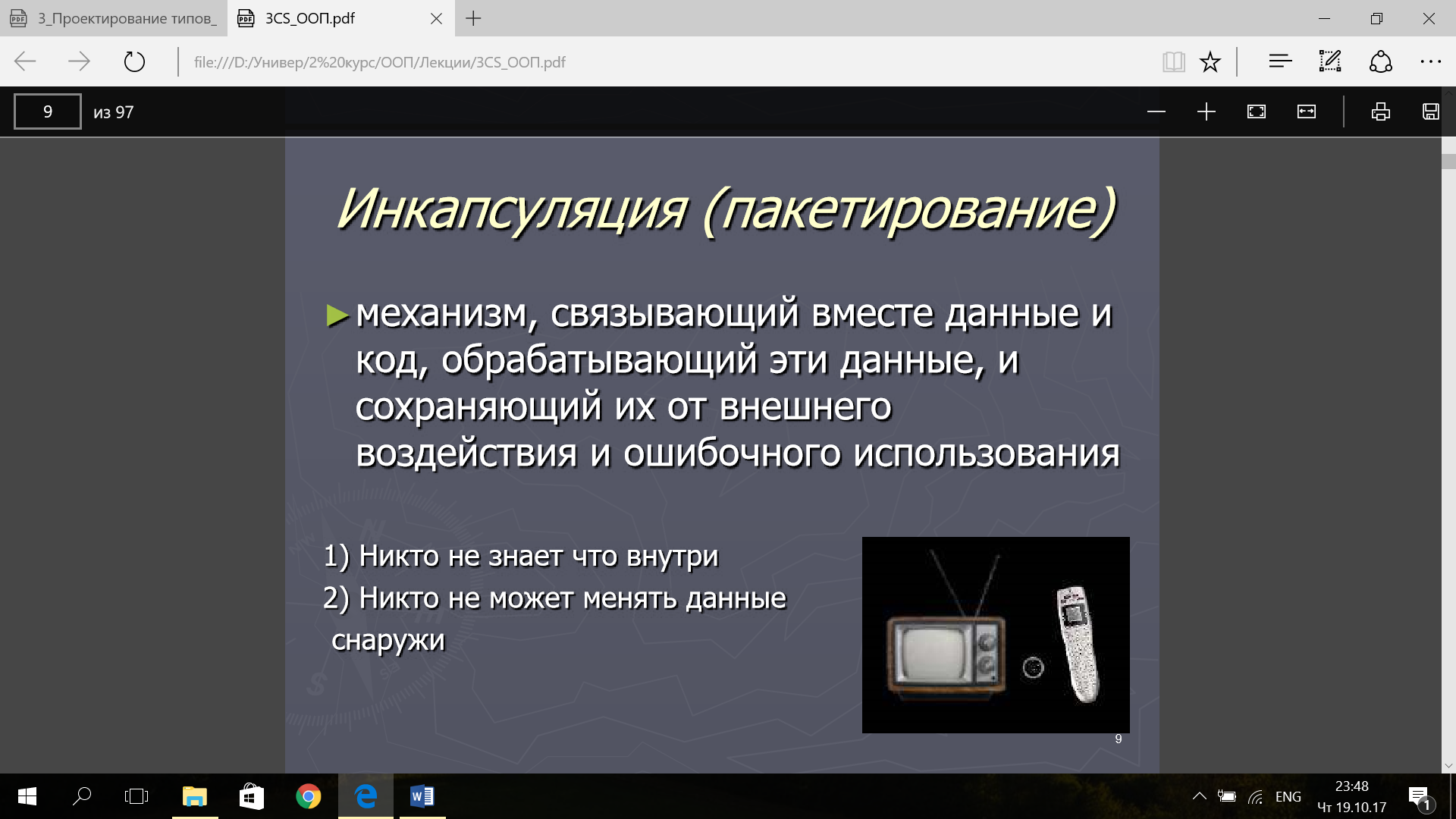
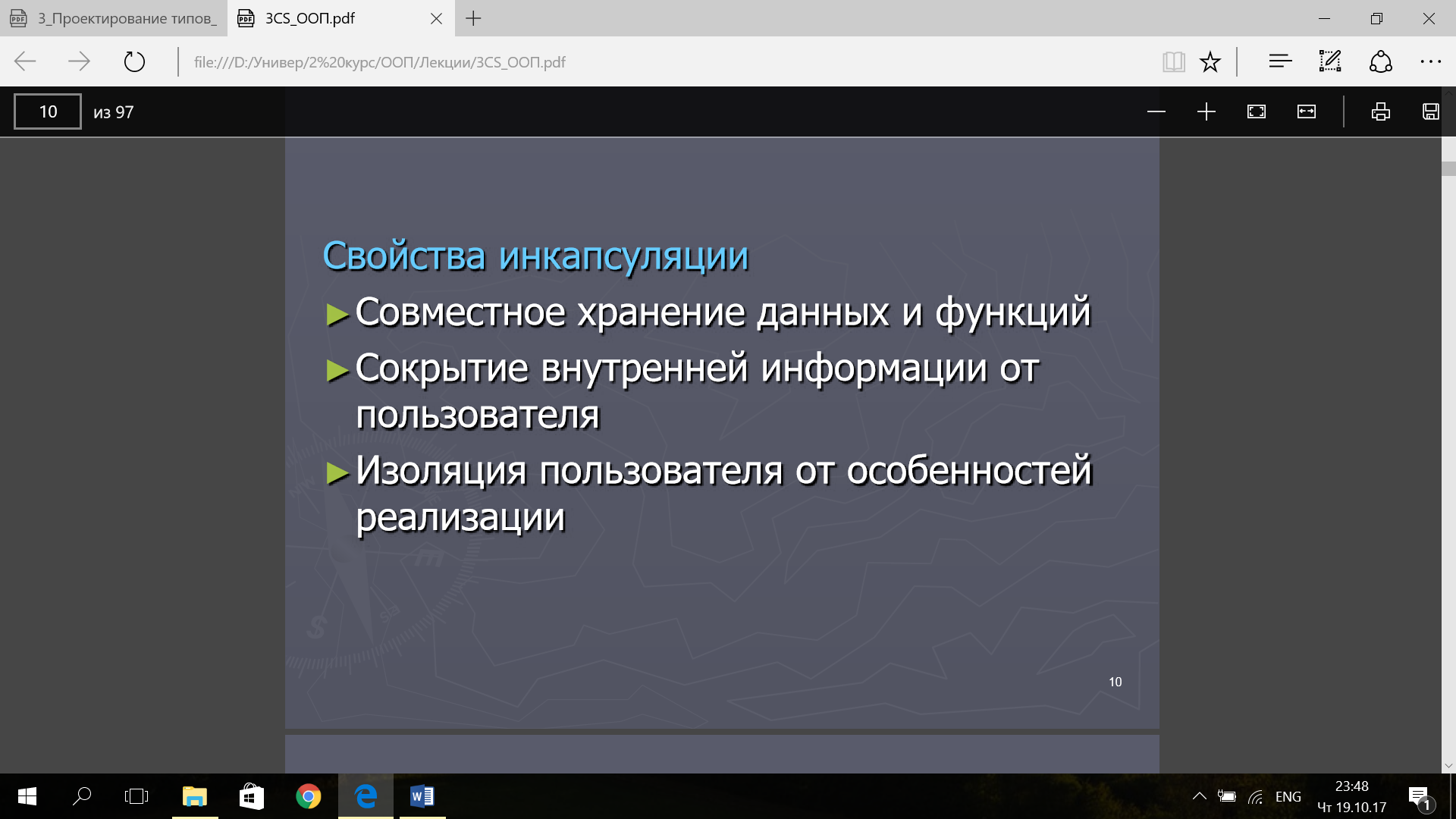
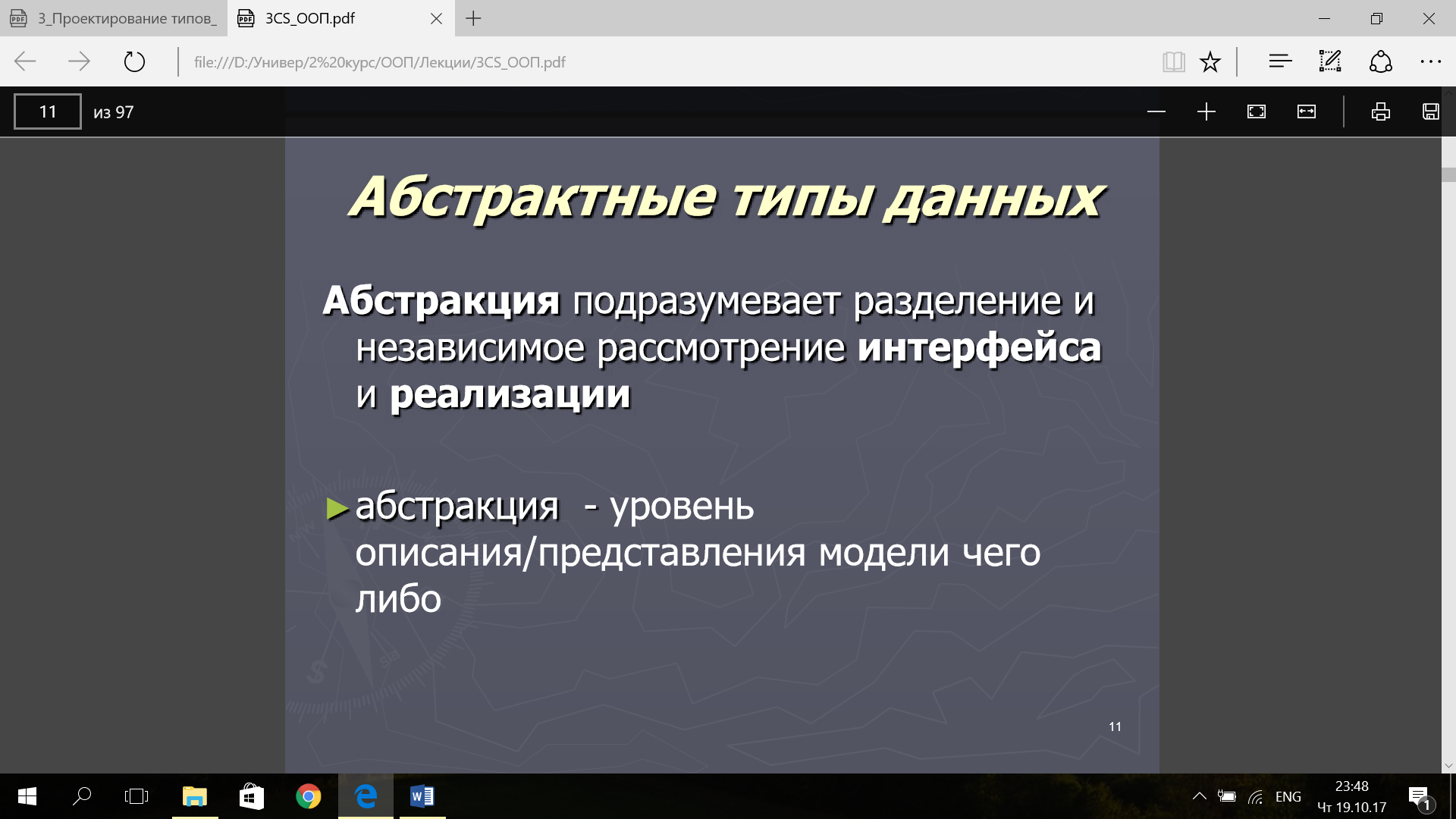
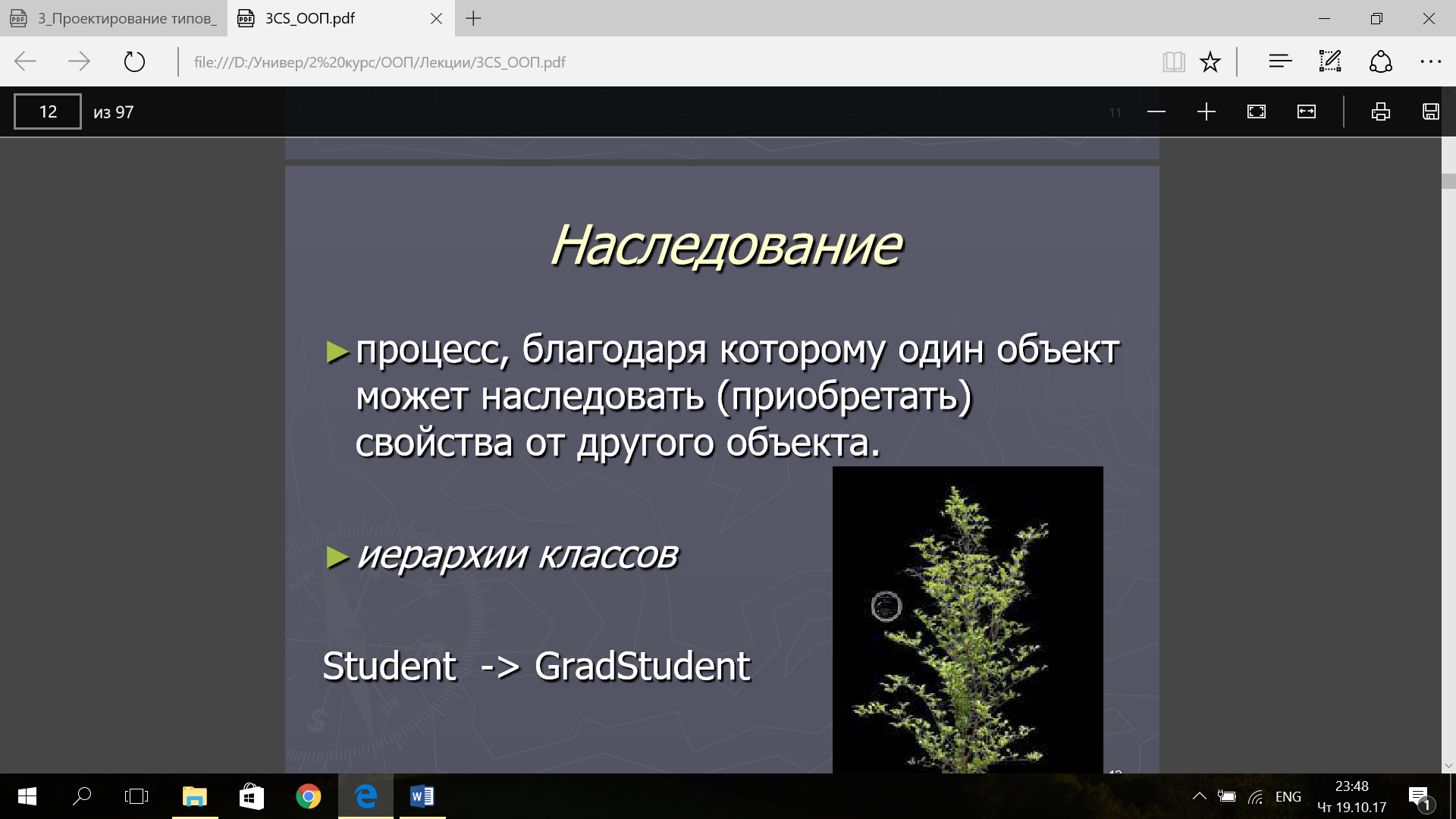
**1,Назовите концепции ООП.**  
Концепции ООП строятся на базе трех основных принципов: 1) инкапсуляция 2)наследование 3)полиморфизм.   
Инкапсуляция – это совмещение структур данных с функциями (методами), манипулирующими этими данными. Фактически, инкапсуляция предполагает размещение в одном месте сразу и структур данных, и методов.  
Наследование – это создание новых классов, которые строятся на базе структур данных и методов уже существующих классов (базовых). В качестве примера рассмотрим windows-окно. Сначала строится класс стандартного windows-окна.   
Полиморфизм – это использование одного имени или идентификатора для метода внутри одной иерархии класса таким образом, чтобы для разных классов этой иерархии этот метод реализовывал различные операции. Функция становится виртуальной, если перед ее объявление поставить спецификатор «virtual».

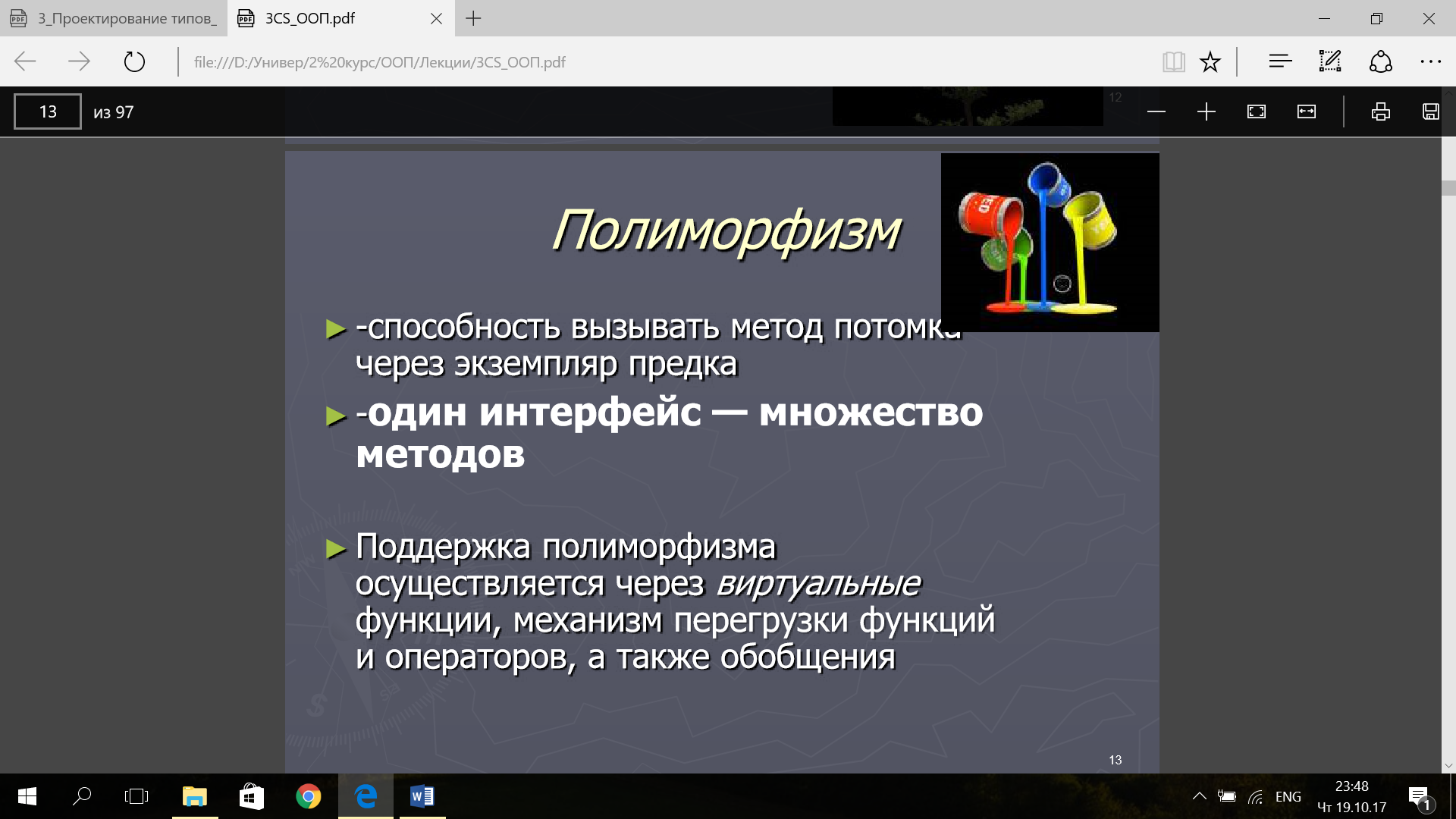












**2.Назовите класс .NET, от которого наследуются все классы?**  
System.Object.

### 3.

|  |  |
| --- | --- |
| Метод | Описание |
| bool Equals() | Сравнивает две ссылки на объекты в период выполнения, чтобы определить, указывают ли они в точности один и тот же объект. Если две переменные ссылаются на один и тот же объект, возвращается *true.* В случае размерных типов (см. о них следующий раздел) этот метод возвращает *true,* если типы переменных идентичны и их значения равны. |
| *int GetHashCodeO* | Возвращает заданный для объекта хзш-код. Хэш-функции используются в реализации класса, когда хэш-код объекта нужно поместить в хэш-таблицу для повышения производительности. |
| Type GetType() | Используется с методами отражения (см. о них главу 16) для получения информации о типе данного объекта. |
| string ToString | Используется по умолчанию для получения имени объекта. Его можно переопределить в производных классах, чтобы они возвращали понятное пользователю текстовое представление объекта. |
|  |  |

4,

|  |  |
| --- | --- |
| Метод | Описание |
| void Finalize() | Вызывается в период выполнения для освобождение ресурсов перед сбором мусора. Этот метод можно вызывать, а можно и не делать этого. Поэтому не помещайте в него подлежащий исполнению код. Это правило выливается в нечто под названием *детерминированное завершение* (deterministic finalization), о котором подробнее см. главу 5. |
| Object MemberwiseClone | Представляет *ограниченную копию* (shallow copy) объекта. Под этим я понимаю копию объекта, содержащую ссылки на другие объекты, но не копии этих объектов. Если ваши классы должны поддерживать *полную копию* (deep copy), которая действительно включает копии объектов, на которые она ссылается, то вам нужно реализовать интерфейс *ICloneable* и самому вручную производить клонирование или копирование. |
|  |  |

5, public class Student

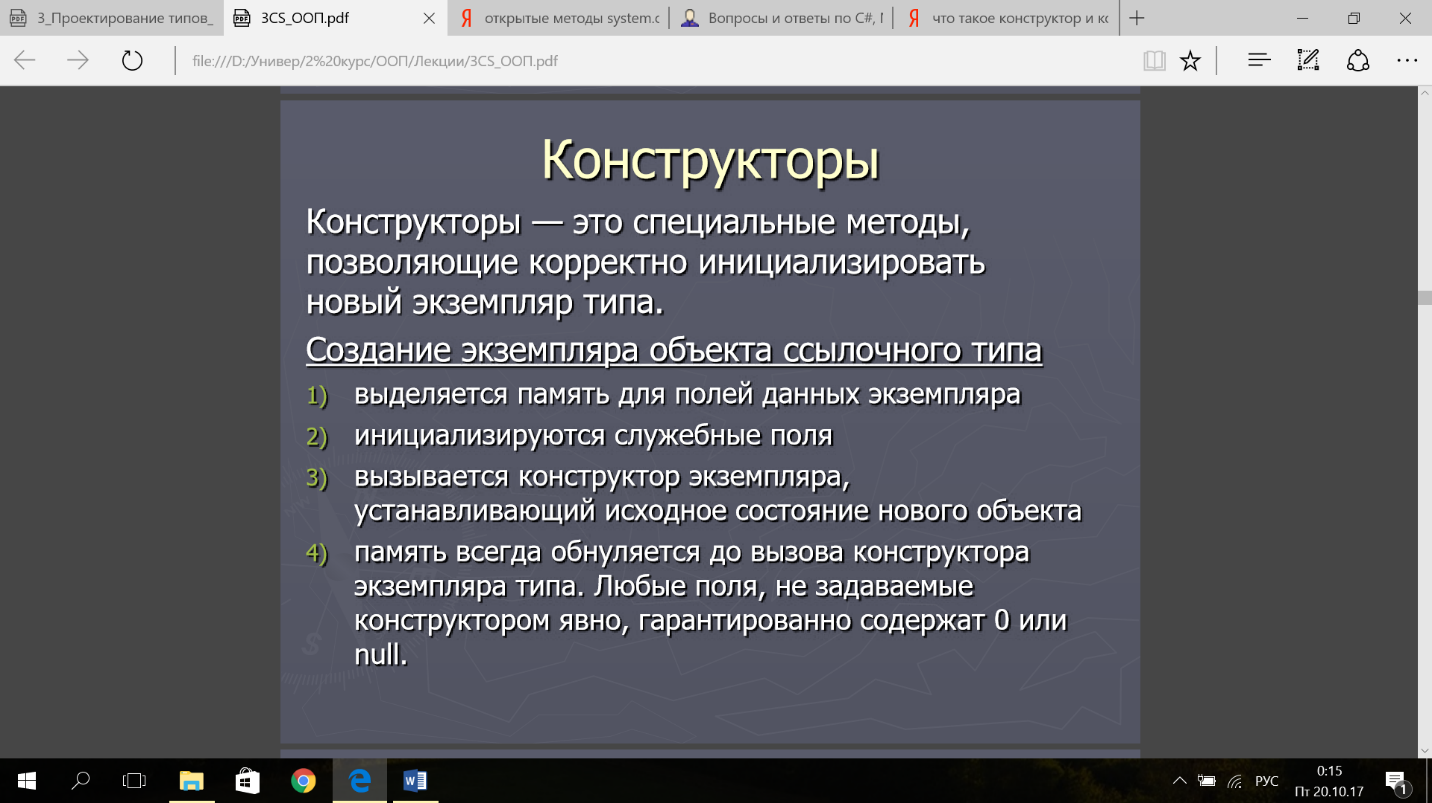
6, ключевое слово **this(**Это ключевое слово представляет ссылку на текущий экземпляр класса)

ключевое слово new(Оператор new создает объект класса и выделяет для него область в памяти.)

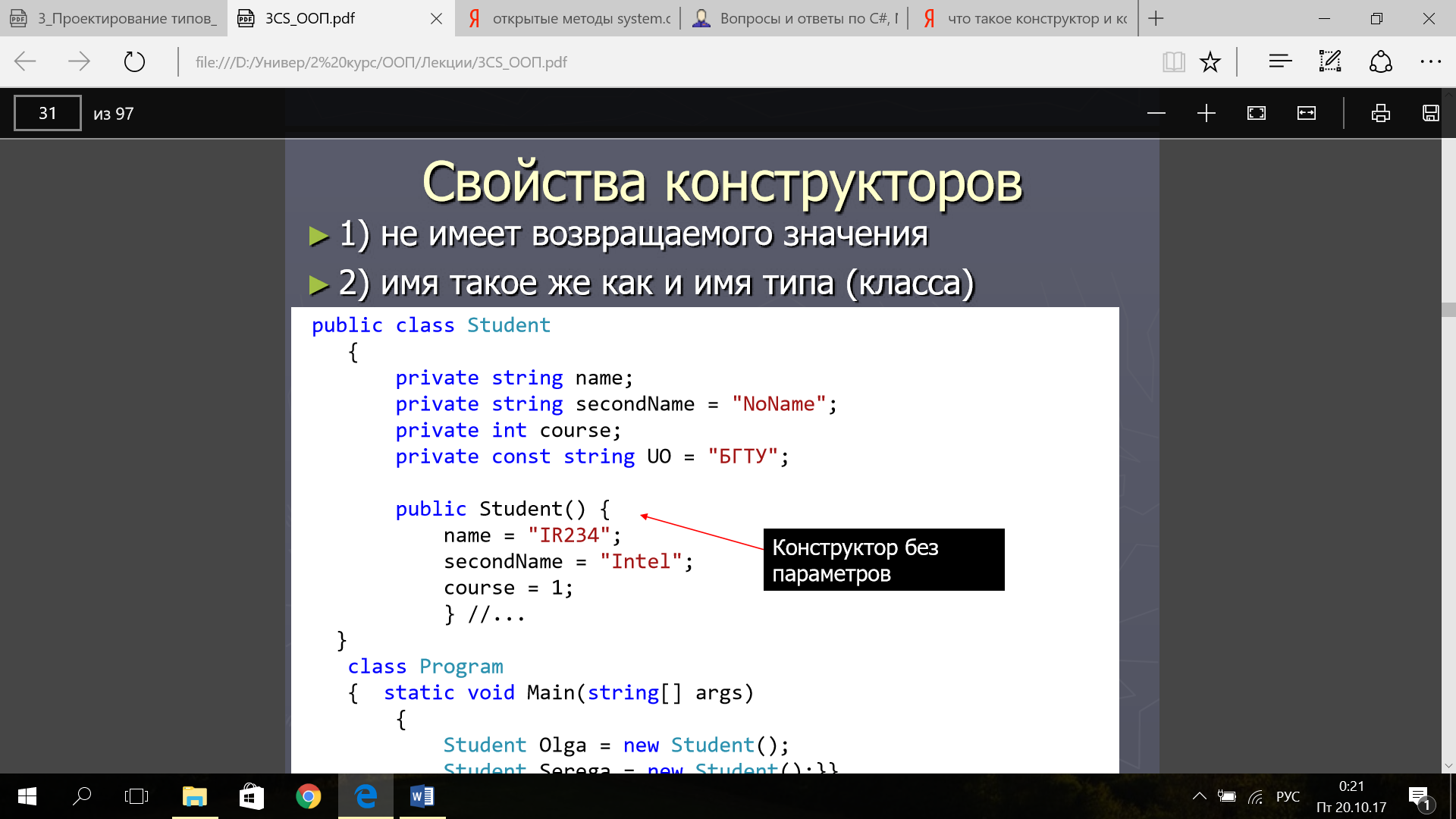
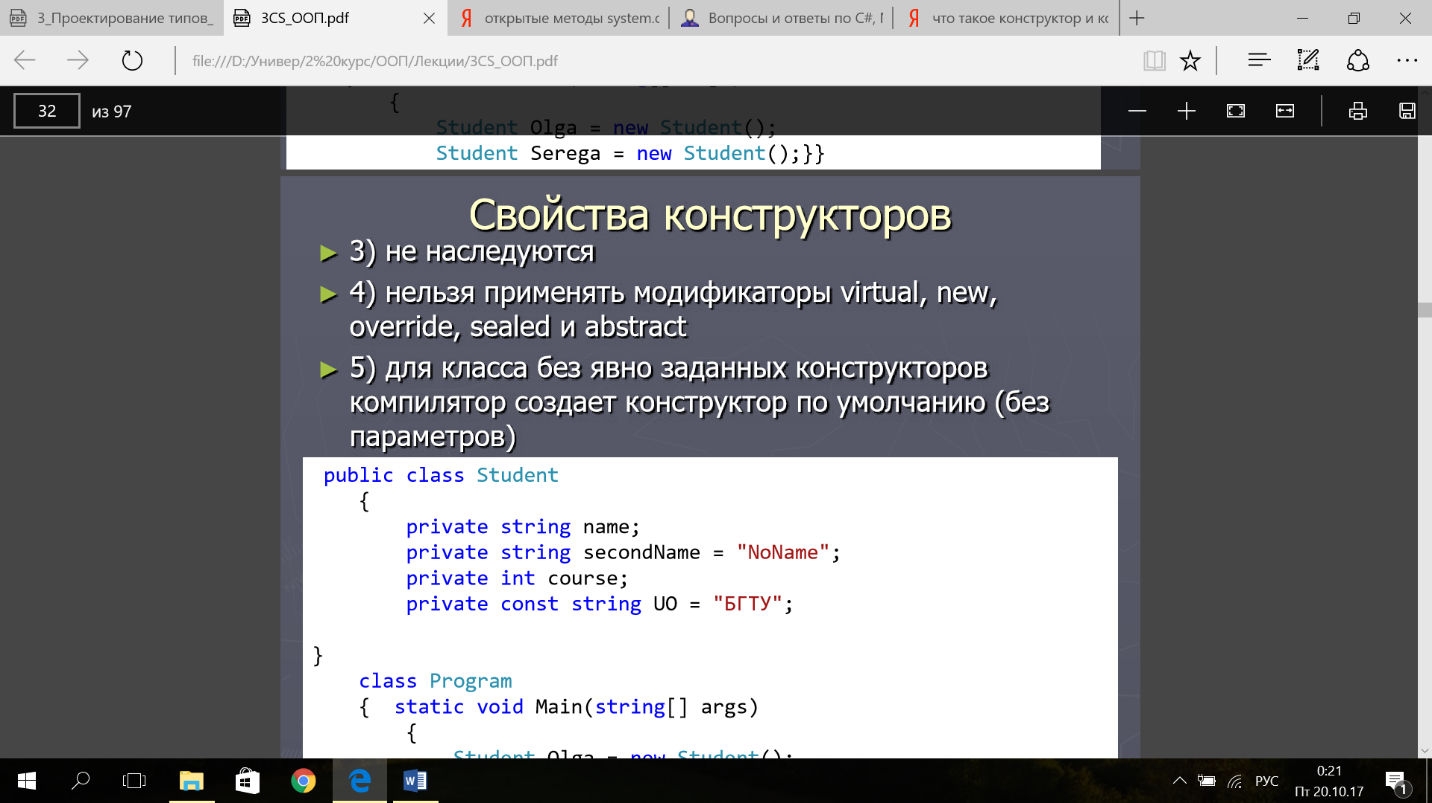
7. **Объект** – структура данных, содержащая описание свойств внешнего объекта программирования.

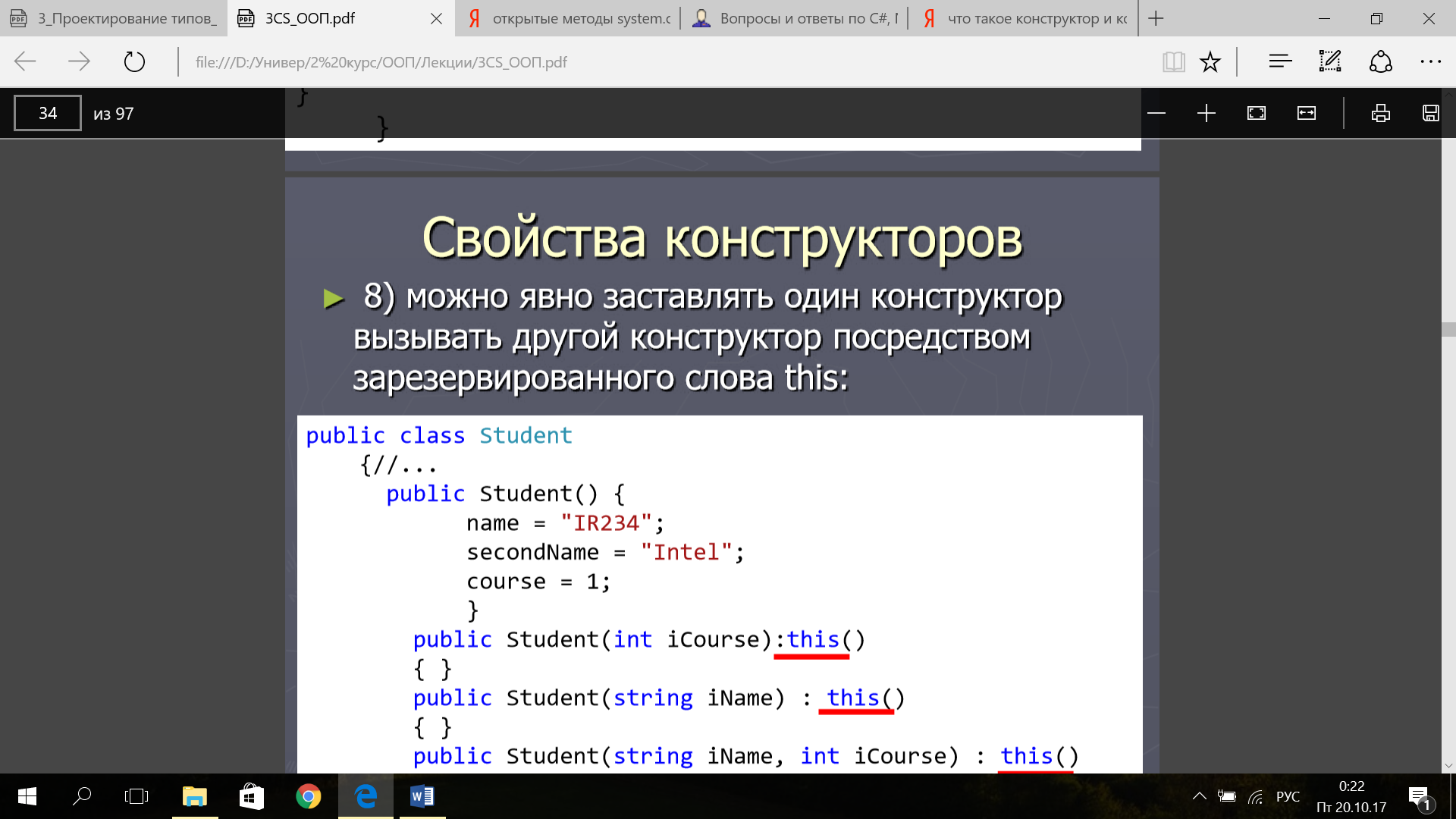
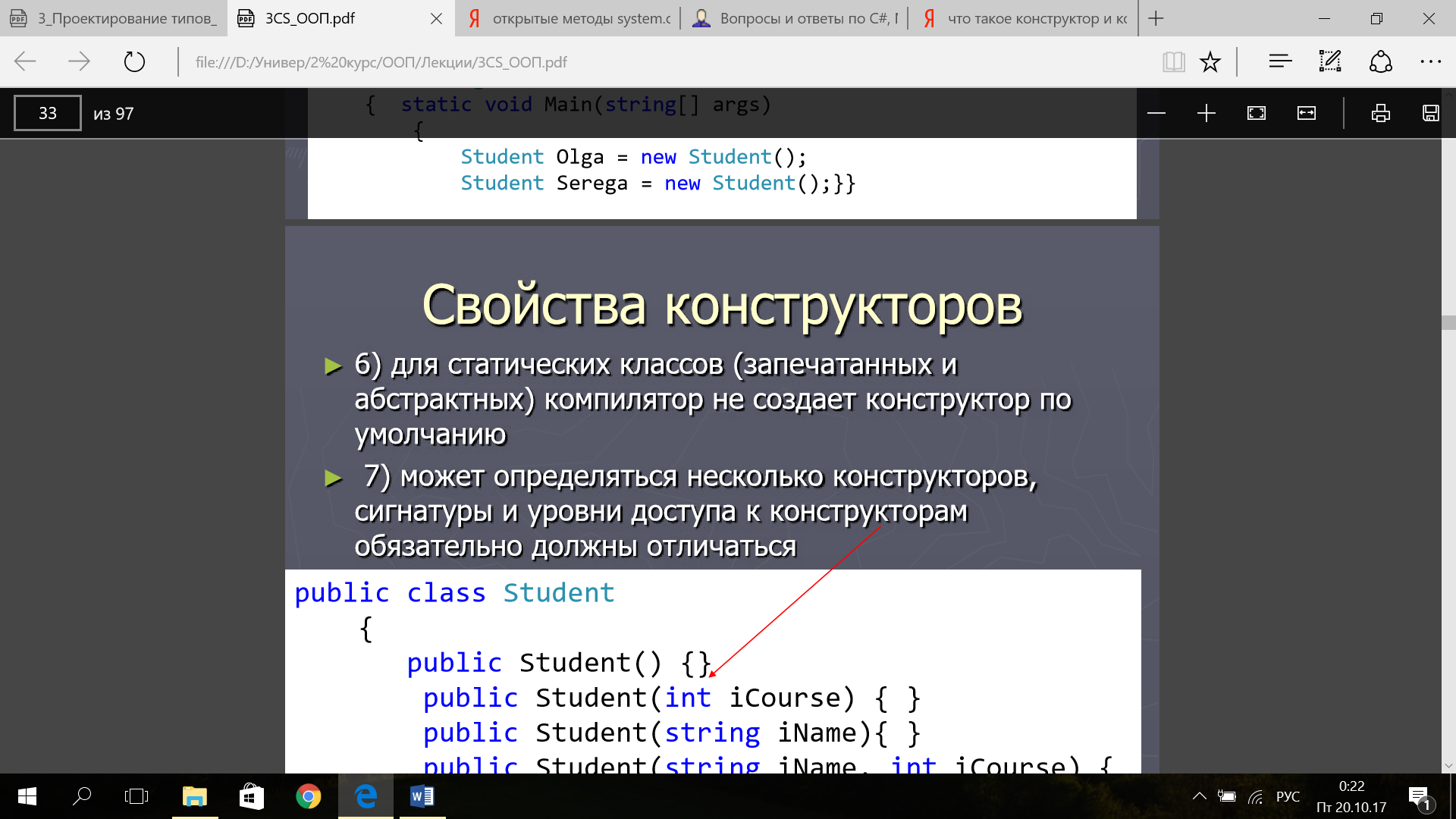
**Класс** – описание структуры объекта и методов работы с ним.

**8**.



9.



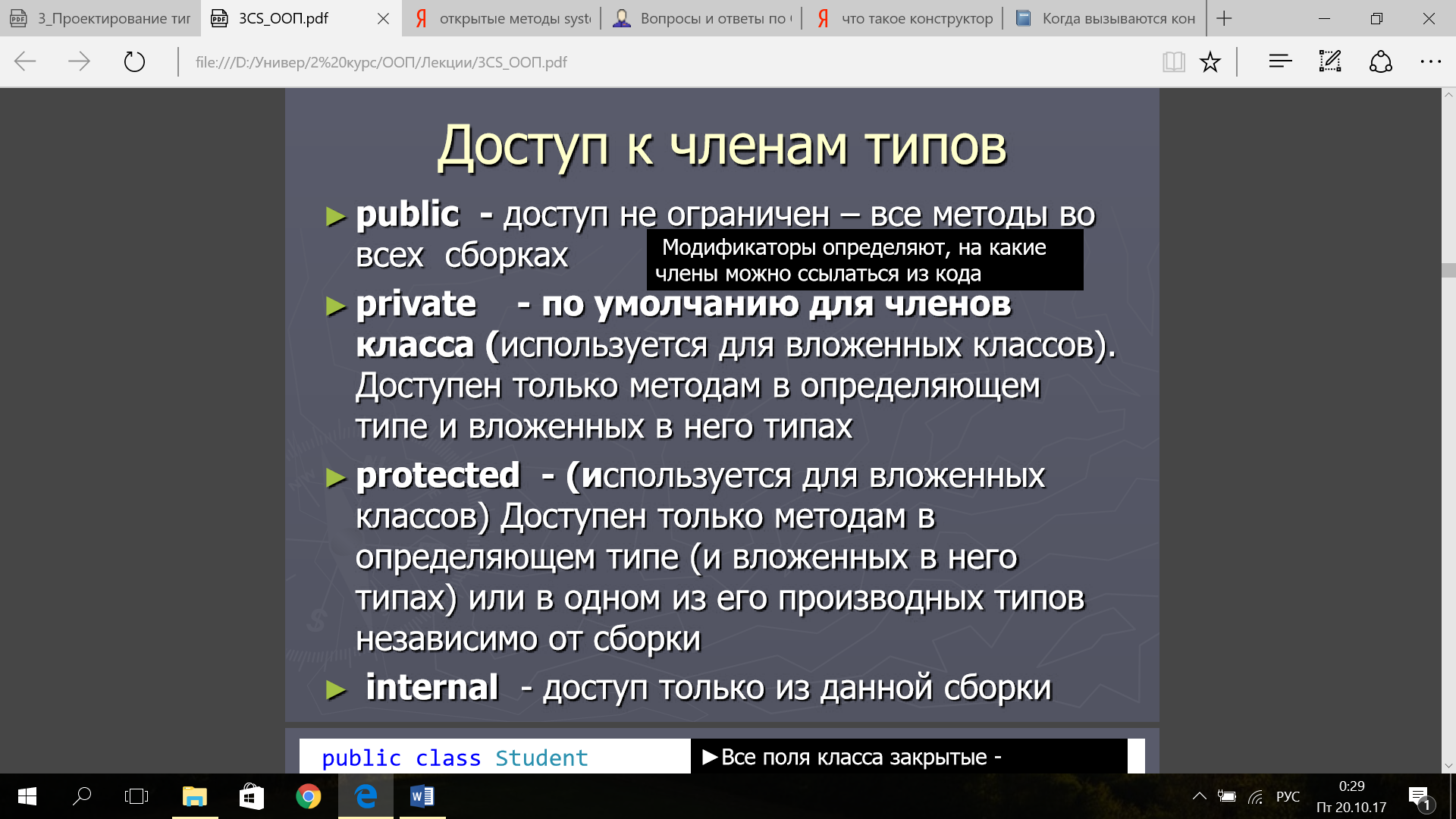


10. Деструктор (destruct - разрушать) - это специальная функция-член класса. Имя деструктора состоит из символа тильда (~) и имени класса.

Пример деструктора: ~my\_Time() { // тело деструктора }

Деструктор решает задачу, обратную задаче конструктора

11. ключевое слово **this(**Это ключевое слово представляет ссылку на текущий экземпляр класса)

13-14. 

15.

