# План на урок за 11-ти клас: Контролирани Атрибути на Клас в Обектно-Ориентираното Програмиране (C#)

## Предмет: Обектно-ориентирано програмиране

Продължителност: 45 минути

Цели на урока:  
1. Учениците да разберат какво представляват контролирани атрибути в ООП в контекста на езика C#.  
2. Да се обясни защо е важно да имаме контрол върху атрибутите на класовете и как се осъществява този контрол в C#.  
3. Да се демонстрират методи за контролиране на достъпа до атрибутите чрез инкапсулация, използвайки get и set в C#.  
4. Да се научат учениците да прилагат автоматични свойства и валидация в set метода.

## Ход на урока

### I. Въведение (5 минути)

1. Цел на урока: Представяне на основната тема и значението на контролираните атрибути в C#.  
2. Повторение: Припомнете накратко концепцията за атрибути и обекти в ООП.  
3. Дискусия: Защо е важно да контролираме достъпа до атрибутите на класовете? Например, какво може да се случи, ако критичен атрибут като balance в банкова сметка бъде променен безконтролно.

### II. Основни понятия (10 минути)

1. Инкапсулация в C#: Представете основите на инкапсулацията като принцип на ООП.  
2. Видове достъп до атрибутите:  
 - Публични полета: Достъпни са отвън, могат да се модифицират без ограничения.  
 - Частни (private) полета: Ограничен достъп чрез модификатора private, за да бъдат защитени.  
3. Свойства (Properties): Обяснете как в C# можем да използваме свойства (get и set методи) за контролиране на достъпа до полетата на класа.

### III. Демонстрация и обяснение (15 минути)

Примерен клас: Представете примерен клас в C#, който използва контролирани свойства.

public class BankAccount  
{  
 private decimal balance; // Частно поле за баланс  
  
 public decimal Balance  
 {  
 get { return balance; } // Getter метод за четене на стойността  
 set  
 {  
 if (value >= 0)  
 balance = value; // Сетър метод за задаване на стойност само ако е положителна  
 else  
 Console.WriteLine("Балансът не може да бъде отрицателен.");  
 }  
 }  
}

Обяснение:  
- Как get позволява четене на стойността на balance.  
- Как set контролира стойността, като позволява само положителни стойности.  
- Каква е ползата от този контрол при атрибути, свързани с важни стойности като balance.  
  
Автоматични свойства: Обяснете автоматичните свойства в C# и кога са подходящи за използване. Например:

public class Product  
{  
 public string Name { get; set; } // Автоматично свойство  
 public decimal Price { get; private set; } // Само за четене отвън  
}

### IV. Практическа задача (10 минути)

Задача: Учениците да създадат клас Product, който има следните атрибути:  
- Name – публично свойство (позволява четене и писане).  
- Price – частно поле с публично свойство, което контролира, че стойността е положителна.  
Инструкции: Насочете учениците да използват get и set за създаване на защитен Price атрибут, който позволява задаване на стойности само ако са по-големи от нула.

public class Product  
{  
 public string Name { get; set; }  
   
 private decimal price;  
 public decimal Price  
 {  
 get { return price; }  
 set  
 {  
 if (value >= 0)  
 price = value;  
 else  
 Console.WriteLine("Цената не може да бъде отрицателна.");  
 }  
 }  
}

### V. Разглеждане на решенията и дискусия (5 минути)

Преглед: Разгледайте някои от решенията на учениците, като наблегнете на коректното използване на get и set.  
Обсъждане: Какви други проверки и ограничения могат да се добавят в set метода за други типове данни?

### VI. Заключение и обобщение (5 минути)

Преговор: Обобщете основните ползи от контролираните атрибути в C#:  
- Безопасност и предсказуемост на данните  
- Увеличена поддръжка на кода чрез добавяне на валидиране.  
Въпроси: Поканете учениците да зададат въпроси и да обсъдите общи примери от реалния свят.

### Домашна работа

Задача: Учениците да разработят клас Student с частни полета Name и Grade. Да се създаде контрол върху Grade, като се позволява само стойности от 2 до 6.

## Допълнителни ресурси за учителя

1. Примери за код: Подгответе различни примери за използване на свойства в C#, като включите валидиране на данни.  
2. Кратко ръководство за свойства в C#: Ако учениците имат нужда от допълнителна помощ, предложете им кратко ръководство за get и set методи в C#.