# Упражнение: Капсулация - Модификатори за достъп

## 1. Увеличение на заплатата

Преструктурирайте (рефактурирайте) проекта с клас Person.

Въвеждаме хора (Person) с техните имена, възраст и заплата. Въвеждаме процент бонус към заплатата на всеки обект person. Обектите Persons, на възраст под 30 получават бонус на половина. Разширяваме Person от предишната задача. Нови полета и методи:

- salary: double
- IncreaseSalary(double bonus)

Използвайте класа например така:

**StartUp.cs**

```
var lines = int.Parse(Console.ReadLine());
var persons = new List<Person>();
for (int i = 0; i < lines; i++)
{
    var cmdArgs = Console.ReadLine().Split();
    var person = new Person(cmdArgs[0],
                            cmdArgs[1],
                            int.Parse(cmdArgs[2]),
                            double.Parse(cmdArgs[3]));

    persons.Add(person);
}
var bonus = double.Parse(Console.ReadLine());
persons.ForEach(p => Console.WriteLine(p.ToString()));
```

### Примери

| Вход | Изход |
|---|---|
| 5<br>Asen Ivanov 65 2200<br>Boiko Borisov 57 3333<br>Ventsislav Ivanov 27 600<br>Asen Harizanoov 44 666.66<br>Boiko Angelov 35 559.4<br>20 | Asen Ivanov get 2640.00 leva<br>Boiko Borisov get 3999.60 leva<br>Ventsislav Ivanov get 660.00 leva<br>Asen Harizanoov get 799.99 leva<br>Boiko Angelov get 671.28 leva |

### Решение

Добавяме ново private поле за заплата salary и променяме конструктора. Добавяме нов метод, който ще променя заплатата с бонус

```
public void IncreaseSalary(double percent)
{
    if (this.age > 30)
    {
        this.salary += this.salary * percent / 100;
    }
    else
    {
        this.salary += this.salary * percent / 200;
    }
}
```

Променяме метода toString() за тази задача.

## 2. Ферма за животни

Трябва да сте запознати с капсулирането вече. За тази задача ще се работи по проект Животинска ферма. Можете да го откриете в AnimalFarm.zip. Той съдържа клас Chicken. Добавете към него няколко полета, конструктор, свойства и метода по ваша преценка, така че да може да се ползва класа по указания в примерите начин. Вашата задача освен това ще бъде да се форматират или скрият членовете на класа, които не са предназначени да се виждат или модифицират извън класа.

### Step 1. Капсулиране на полетата

Полетата трябва да бъдат private. Оставянето на полетата, отворени за промяна извън класа, е потенциално опасно. Направете всички полета в класа Chicken private. В случай, че стойността на полето е необходимо другаде, използвайте getters за достъп до него.

### Step 2. Подходящо валидиране на данните (вижте как в упражнението за валидация)

Валидирайте името на обектите от клас Chiken (не може да е null, празно или да съдържа само интервали). В случай на невалидно име, да се върне изключение със следното съобщение "Name cannot be empty."

Валидирайте свойството age, с минималната и максималната допустими стойности. В случай на невалидна възраст, върнете изключение със съобщение "Age should be between 0 and 15."

Не забравяйте да обработвате правилно вероятно получените изключения.

### Step 3. Направете класове да имат валидно начално състояние

Наличието на getters и setters е безполезно, ако всъщност не ги използвате. Конструкторът на класа Chicken променя полетата директно, което е погрешно, когато са налице подходящи механизми за валидиране на входните данни. Променете конструктора за да разрешите този проблем.

### Step 4. Скрийте вътрешната логика

Ако метод е предназначен да се използва само от наследяващите класове или вътрешно да извърши някакво действие, няма смисъл да бъде публичен. Методът CalculateProductPerDay() се използва от productPerDay(), който е public

getter. Това означава, че методът безопасно може да бъде скрит във вътрешността на класа Chicken като се декларира като private.

## Примери

| Вход | Изход |
|---|---|
| Mara<br>10 | Chicken Mara (age 10) can produce 1 eggs per day. |
| Mara<br>17 | Age should be between 0 and 15. |