**Основни понятия в ООП**

1. **Обектно-ориентираното програмиране (ООП)** е програмна парадигма, базирана на концепцията за „обекти“, които могат да съдържат данни и код: данни под формата на полета (често известни като атрибути или свойства) и код под формата на процедури (често известни като методи).
2. **Клас-**дефинициите за формата на данните и наличните процедури за даден тип или клас обект; може също така да съдържа самите данни и процедури (известни като методи на класове), т.е. класовете съдържат членове с данни и функции на членове.
3. **Обекти-** инстанции на класове, създадени със специфично дефинирани данни. Обектите могат да съответстват на обекти от реалния свят или абстрактни обекти. Когато класът е дефиниран първоначално, описанието е единственият обект, който е дефиниран.
4. **Състояния(states)-** Състоянието е модел на поведенчески дизайн, който позволява на обект да промени поведението си, когато вътрешното му състояние се промени. Моделът извлича поведение, свързано със състоянието, в отделни класове на състояние и принуждава оригиналния обект да делегира работата на инстанция от тези класове, вместо да действа
5. **Поведения(behaviours)-**Поведенията са задачите, които даден обект изпълнява. Атрибутите на човек, например, включват неговата възраст, име и височина, докато поведението му включва факта, че човек може да говори, да тича, да ходи и да яде.

1. **Член-данни(data members)-** Променливите, които са декларирани във всеки клас чрез използване на фундаментални типове данни (като int, char, float
2. **Методи на клас.Методи(methods)-** Методът е кодов блок, който съдържа поредица от изрази. Програмата кара операторите да бъдат изпълнени чрез извикване на метода и указване на всички необходими аргументи на метода. В C# всяка изпълнена инструкция се изпълнява в контекста на метод. Методите се декларират в клас, структура или интерфейс чрез указване на нивото на достъп като public или private, незадължителни модификатори като abstract или sealed, върнатата стойност, името на метода и всички параметри на метода. Тези части заедно са сигнатурата на метода. Параметрите на метода са оградени в скоби и са разделени със запетаи. Празните скоби показват, че методът не изисква параметри.
3. **Инстанциране и Инстанция**:

Създаването на обект от вече дефиниран клас -наричаме **инстанциране** (instantiation). **Инстанция** (instance) е фактическият обект, който се създава от класа по време на изпълнение на програмата. Всеки обект е **инстанция**на конкретен клас. Тази инстанция се характе­ризира със **състояние** (state) – множество от стойности, асоциирани с атрибутите на класа. В контекста на така въведените понятия, обектът се състои от две неща: моментното състояние и поведението, дефинирано в класа на обекта. Състоянието е специфично за инстанцията (обекта), но поведението е общо за всички обекти, които са представители на този клас.

1. **Полета(fields)-** те са променливи, декларирани в класа (някъде се срещат като **член-променливи**). В тях се пазят данни, които отразяват състоянието на обекта и са нужни за работата на методите на класа. Стойността, която се пази в полетата, отразява конкретното състояние на дадения обект, но съществуват и такива полета, наречени **статични**, които са общи за всички обекти.
2. **Свойства (properties):**

така наричаме характеристиките на даден клас. Обикновено стойността на тези характеристики се пази в полета. Подобно на полетата, свойствата могат да бъдат притежа­вани само от конкретен обект или да са споделени между всички обекти от тип даден клас.

**Ресурси**:

[**https://introprogramming.info/intro-csharp-book/read-online/glava14-definirane-na-klasove/**](https://introprogramming.info/intro-csharp-book/read-online/glava14-definirane-na-klasove/)

[**https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/methods**](https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/methods)

[**https://en.wikipedia.org/wiki/Object-oriented\_programming**](https://en.wikipedia.org/wiki/Object-oriented_programming)

[**https://overcoded.dev/posts/BC-51**](https://overcoded.dev/posts/BC-51)

[**https://www.techopedia.com/definition/25589/class-members-c-sharp**](https://www.techopedia.com/definition/25589/class-members-c-sharp)

[**https://www.oreilly.com/library/view/hands-on-object-oriented-programming/9781789617726/ca7ee1ce-8d09-4e97-a220-feb6c10efd10.xhtml**](https://www.oreilly.com/library/view/hands-on-object-oriented-programming/9781789617726/ca7ee1ce-8d09-4e97-a220-feb6c10efd10.xhtml)