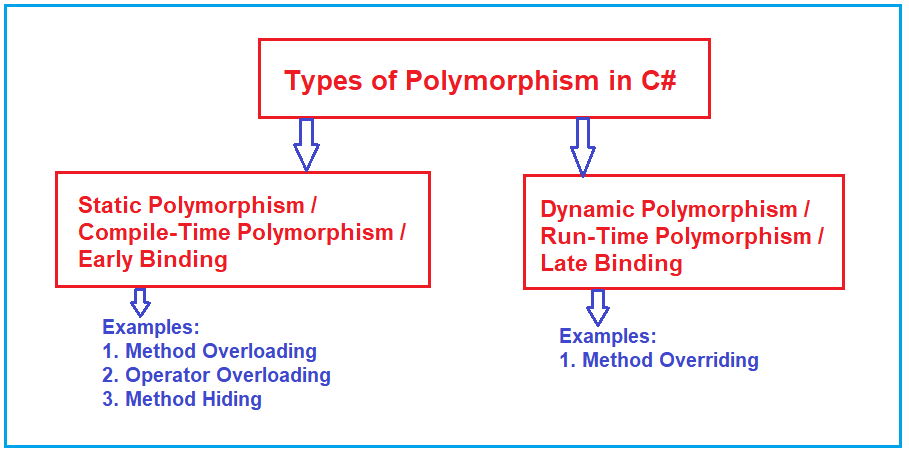
**What is Polymorphism in C#?**

Polymorphism is one of the fundamental OOPs concepts and is a term used to describe situations where something takes various roles or forms.

In the programming world, these things can be operators or functions.

Polymorphism is a concept by which we can perform a single task in different ways.



##### ****What is Compile-Time Polymorphism in C#?****

In **static polymorphism**, the behaviour of a method is decided at compile time.

That means the C# compiler binds method calls with method definition/body during compilation time only.

Therefore, this type of polymorphism is also called **compile-time polymorphism** in C#.

As the binding (the link between the function call and function definition) is performed at compile time, it is also known as **early binding**.

This happens in the case of Method Overloading because, in this case, each method will have a different signature, and based on the method call, we can easily recognize the method which matches the method signature.

##### ****What is Runtime Polymorphism in C#?****

In dynamic polymorphism, the behaviour of a method is decided at runtime, therefore, the CLR (Common Language Runtime) binds the method call with method definition/body at runtime and invokes the relevant method during runtime when the method is called.

The function call is bounded to the class at the time of compilation, if the function is going to be executed from a different class at run-time rather than the class bounded at compilation time, then it is called Run-Time Polymorphism.

This happens in the case of Method Overridingbecause, in the case, we have multiple methods with the same signature i.e. Parent Class and the Child class have the same method implementation. So, in this case, we will be able to know at runtime from which class the method is going to be executed.

It is also called Dynamic Polymorphism or Late Binding as at Run-time we will be able to know from which class the method is going to be executed.