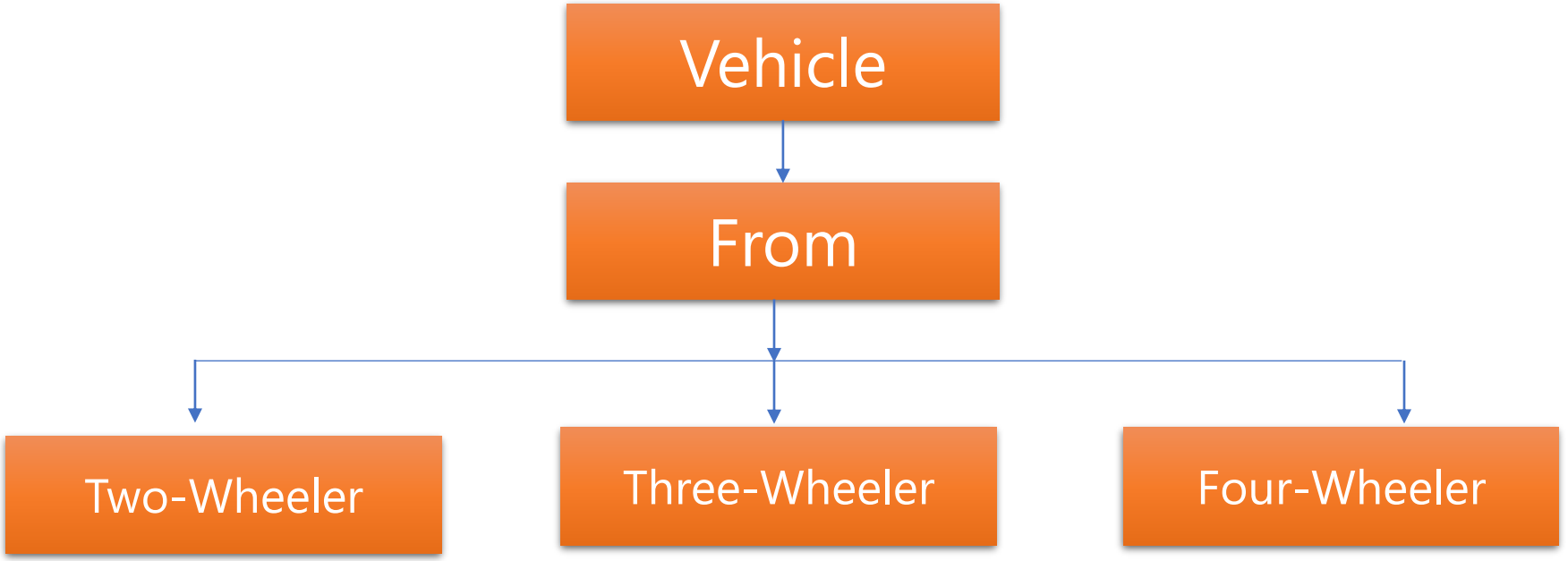
# Polymorphism in C#

**What is Polymorphism in C#?**

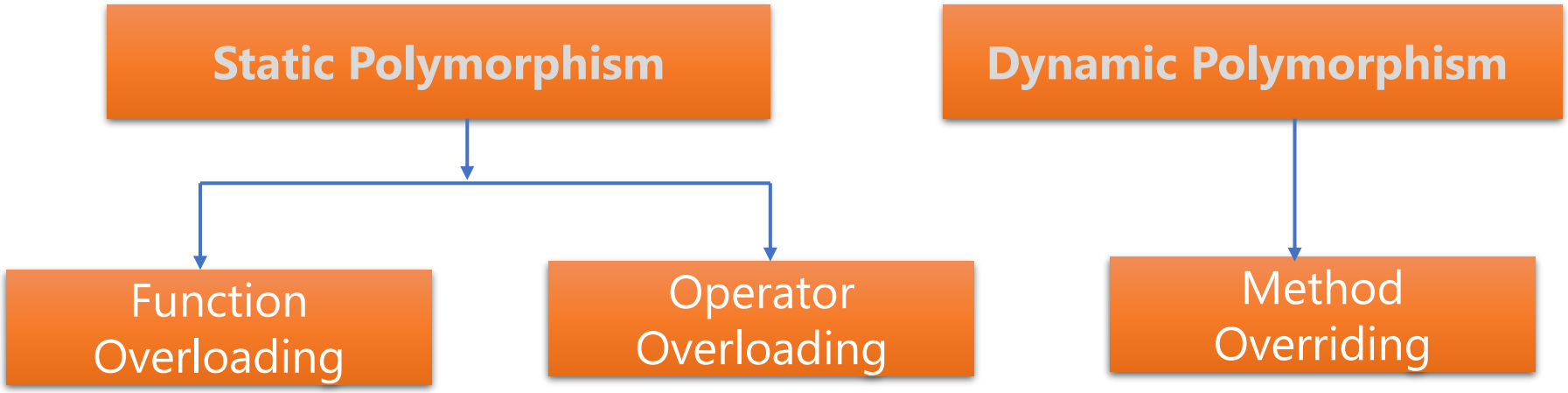
The word polymorphism is derived from the Greek word, where Poly means many and morph means faces/ behaviors. So, the word polymorphism means the ability to take more than one form. Please have a look at the following diagram. As shown in the below diagram, the vehicle is something that has various forms; two-wheeler, three-wheeler, and four-wheeler, and so on. So this is one example of polymorphism.




**Types of Polymorphism in C#**

**Static Polymorphism / Compile-Time Polymorphism / Early Binding**

**Dynamic Polymorphism / Run-Time Polymorphism / Late Binding**




**What is Method Overloading or Function Overloading in C#?**

**Method Overloading in C#** allows a class to have multiple methods with the same name but with a different signature. The functions or methods can be overloaded based on the number, type (int, float, etc), order, and kind (Value, Ref or Out) of parameters

**Method(X)** **Method(X,Y)** **Method(X,Y,Z)**

**1- Method name is same**
**2- Number, Type and Order of parameters is different**

**What is Method Overriding in C#?**

The process of re-implementing the superclass non-static and non-private method in the subclass with the same signature is called Function Overriding or Method Overriding in C#. The same signature means the name and the parameters should be the same. The implementation of the subclass overrides (i.e. replaces) the implementation of the superclass method

## When do we need to override a method in C#?

If the superclass method logic is not fulfilling the sub-class business requirements, then the subclass needs to override that method with the required business logic.

## How can we override a parent class method under child class in C#?

**Syntax:**

**Class1:**
**Public virtual void show(){} //virtual function (overridable)**
**Class2: Class1**
**Public override void show(){} //overriding**

## Using the base keyword to call the parent class methods in C#:

**base.show()**

**What is the difference between Method Overloading and Method Overriding in C#?**

| Method Overloading in C# | Method Overriding in C# |
|---|---|
| It is an approach of defining multiple methods with the same name but with a different signature. | It is an approach of defining multiple methods with the same name and with the same signature. |
| Overloading a method can be performed within a class or within the child classes also. | Overriding of methods is not possible within the same class it must be performed under the child classes. |
| To overload a parent class method under the child class, the child class does not require permission from the parent. | To override a parent class method under the child class, first, the child class requires explicit permission from its parent. |
| This is all about defining multiple behaviors to a method. | This is all about changing the behavior of a method. |
| Used to implement static polymorphism. | Used to implement dynamic polymorphism. |
| This is a code refinement technique. | This is a code replacement technique. |
| No separate keywords are used to implement function overloading. | Use the **virtual** keyword for the base class function and **override** keyword in the derived class function to implement function overriding. |

**What is Method Hiding in C#?**

When we use the **new** keyword to hide a base class member, then it is called **Method Hiding in C#**. We will get a compiler warning if we miss the new keyword. This is also used for re-implementing a parent class method under child class. Re-implementing parent class methods under child classes can be done using two different approaches, such as

**1.Method Overriding**

**2.Method Hiding**

In the first case, we re-implement the parent class methods under child classes with the permission of parent class because here in parent class the method is declared as **virtual** giving permission to the child classes for overriding the methods using the **override** modifier.

In the 2nd approach, we re-implement the method of parent class even if those methods are not declared as virtual that is without parent permission we are re-implementing the methods.

**The syntax for Method Hiding in C#:**

Please have a look at the following image to understand the syntax of Function Hiding in C#.

```
Public Class1 :
Public Void Display();

Public Class2:Class1
Public New Void Display();
```