What is class?
A class is **a user-defined blueprint or prototype from which objects are created**. Basically, a class combines the fields and methods(member function which defines actions) into a single unit. In C#, classes support polymorphism, inheritance and also provide the concept of derived classes and base classes.

# Introduction

C# supports two types of class methods, static methods, and non static methods. Any normal method is a non static method.

A static method in C# is a method that keeps only one copy of the method at the Type level, not the object level. That means, all instances of the class share the same copy of the method and its data. The last updated value of the method is shared among all objects of that Type.

Static methods are called by using the class name, not the instance of the class.

The Console class and its Read and Write methods are an example of static methods. The following code example calls Console.WriteLine and Console.ReadKey methods without creating an instance of the Console class.

```
class Program
{
    public static void withoutObj()
    {
        Console.WriteLine("Hello");
    }
     static void Main()
    {
        Program. withoutObj();
        Console.ReadKey();
    }
}
```

## Using Static Method

Usually we define a set of data members for a class and then every object of that class will have a separate copy of each of those data members. Let's have an example.

```
class Program
  {
      public int myVar;  //a non-static field
        static void Main()
      {
          Program p1 = new Program();  //a object of class
          p1.myVar = 10;
          Console.WriteLine(p1.myVar);
          Console.ReadKey();
      }
  }
```

In the above example, myVar is a non-static field so to use this field we first need to create the object of that class. On the other hand, static data is shared among all the objects of that class. That is, for all the objects of a class, there will be only one copy of static data. Let's have an example.

```
class Program
 {
     public static int myVar;  //a static field
       static void Main()
     {
         //Program p1 = new Program();  //a object of class
         myVar = 10;
         Console.WriteLine(myVar);
         Console.ReadKey();
     }
 }
```

In the above we don't have an object of the class to use that field since the field is static.

If you create your own class and you think only one copy of the data (method) is needed among all instances of the class, you can create your own static method.  Learn more here: Static Class  and Static Class Members In C#

Notable Points here are:

1. A static method can be invoked directly from the class level
2. A static method does not require any class object
3. Any main() method is shared through the entire class scope so it always appears with static keyword.
4. **A class that is static can have static methods / members only.**

Data is not stored in a static class, because it lives in the entire life of your application, Memory footprints will be not used efficiently.