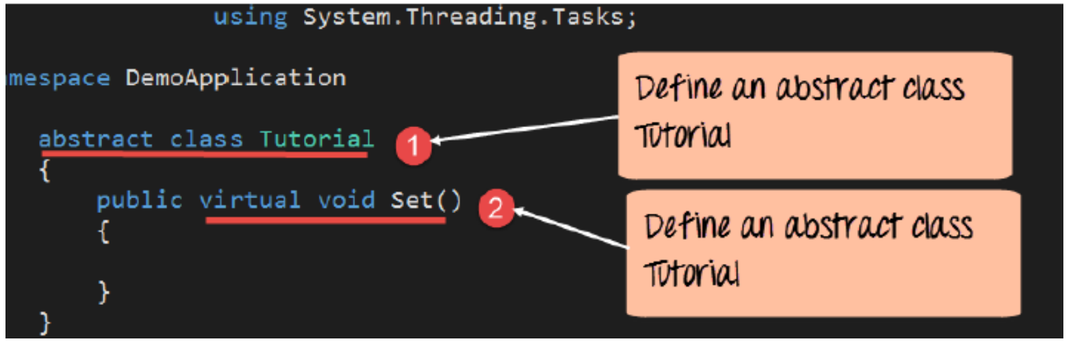
OOP terminology in C#

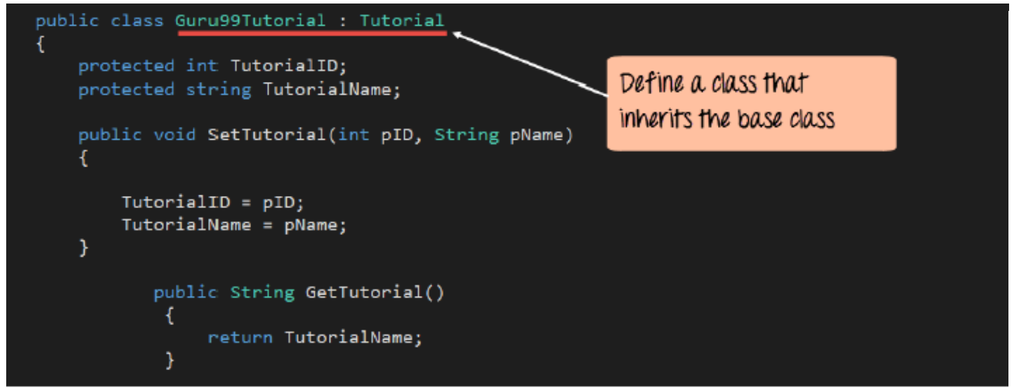
What is Abstract Class in C#?
Abstract Class can never be instantiated and is marked by the keyword abstract. An abstract class contains zero or more abstract methods in it. Abstract class acts as a base class and is designed to be inherited by subclasses that either implement or either override its method.
Let's learn abstract class in C# with example given below. Below is the definition of a class called 'Animal.' When the 'Animal' class is defined, there is nothing known about the animal, whether it is a dog or a cat. The method called description is just a generic method defined for the class.

Now when it is known what exactly the Animal is going to be, we create another class which inherits the base class. If we know that the animal is in fact a Dog, we create Dog class which inherits the main base class.
The key difference here is that the Dog class cannot change the definition of the Description method of the Animal class. It has to define its own C# abstract method called Dog-Description. This is the basic concept of C# abstract classes.

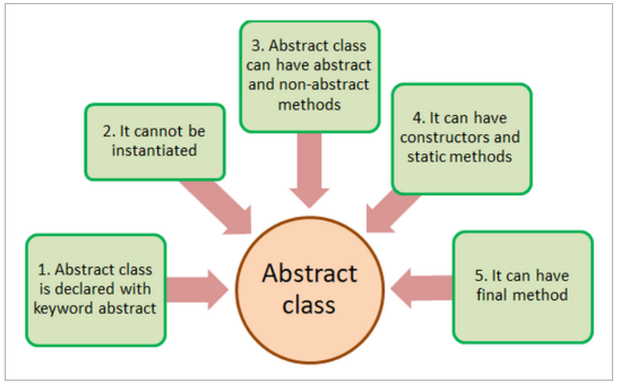
Step 1) As a first step, let's create an abstract class. The class will be called Tutorial and will just have one method. All the code needs to be written in the Program.cs file.



Code Explanation:-
1. We first define the abstract class. Note the use of the abstract keyword. This is used to denote that the class is an abstract class.
2. Next, we are defining our method which does nothing. The method must have the keyword called virtual. This means that the method cannot be changed by the child class. This is a basic requirement for any abstract class.
Step 2) Now let's add our child class. This code is added to the Program.cs file.



An abstract class in C sharp is a base class that has the very basic requirements of what a class should look like. It is not possible for the child class to inherit the methods of the base class.




```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace DemoApplication
{
 abstract class Tutorial
 {
  public virtual void Set()
 {

 }
}
 class Guru99Tutorial:Tutorial
 {
  protected int TutorialID;
  protected string TutorialName;

  public void SetTutorial(int pID,string pName)
  {
   TutorialID=pID;
   TutorialName=pName;
  }

  public String GetTutorial()
  {
   return TutorialName;
  }

 static void Main(string[] args)
 {
  Guru99Tutorial pTutor=new Guru99Tutorial();

  pTutor.SetTutorial(1,".Net");

  Console.WriteLine(pTutor.GetTutorial());

  Console.ReadKey();
 }
}
}
```