2.2 counter class

program.cs

```
using System;

using System.Diagnostics.CodeAnalysis;

using System.Diagnostics.Metrics;


namespace counterclass

{

  public class program

  {

    private static void PrintCounter(counter[] counters)//we call the counter.cs into a new variable counters

    {

      foreach (counter c in counters)

      {

        //Console.WriteLine(c.Name + " is " +c.Tick);

        Console.WriteLine("{0} is {1}", c.Name, c.Tick); //replace {0},{1} and c.Name,c.Tick


      }

    }


    public static void Main(string[] args)

    {

      counter[] myCounters = new counter[3];

      myCounters[0] = new counter("Counter 1"); //first counter display as gievn

      myCounters[1] = new counter("Counter 2"); //second counter display as given
```

```
        myCounters[2] = myCounters[0]; //when myCounters[2] it will call back to
counter[0]


        for (int i = 0; i <= 9; i++) //set i till reaches 9 then display next value above the set i
value
        {
            myCounters[0].increment(); //use the increment value from counter.cs to
increase the i value
        }
        for (int i = 0; i <= 14; i++) //set i till reaches 14 then display next value above the set
i value
        {
            myCounters[1].increment(); //use the increment value from counter.cs to
increase the i value
        }


        PrintCounter(myCounters); //print the console.WriteLine from the line above
        Console.ReadLine();//read the myCounter input
        myCounters[2].Reset(); //use the reset function from counter.cs
        PrintCounter(myCounters); //print again the PrintCounter
        Console.ReadLine(); //read the myCounter input


    }
  }
}


//printing counter values
//PrintCounter(myCounters);
//prints the current state of all counters.
```

```
//Since myCounters[0] and myCounters[2] are the same object,
//they will have the same Tick value (10). myCounters[1] will
//have a Tick value of 15.

//PrintCounter(myCounters);
//prints the current state of all counters.
//Since myCounters[0] and myCounters[2] are the same object,
//they will have the same Tick value (10). myCounters[1] will have a Tick value of 15.

//Printing the Counter Values Again:
//PrintCounter(myCounters); prints the updated state.
//Now, myCounters[0] and myCounters[2] will show a Tick value of 0,
//while myCounters[1] remains at 15.Printing the Counter Values Again:
```

Counter.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace counterclass
{
   public class counter
   {
      private int _count;
```

```
    private string _name; //declare as private class here



    public counter(string name) //intialize the variable _name and _count set value
    {
      _name = name;
      _count = 0;


    }
    public void increment() //this code of line helps to do increment to our counter +1 per tick
    {
      _count++; //+1
    }
    public void Reset() //this code meanwhile helps reset the counter to 0
    {
      _count = 0;//set 0
    }
    public string Name //declare the variable as public class so we are able to call the variable
    {
      get
      {
        return _name; //call the variable _name to use for set value below
      }
      set
      {
        _name = value; //transfer the value to our _name
      }
```

```
    }
    public int Tick
    {
      get
      {
          return _count; //per tick is based on our increment in _count++ so now we call the variable again to be used
      }
    }


  }

}
```

Output for the program