COS20007: Object Oriented Programming

Pass Task 3.1: Clock Class with your own hour format

Show Wai Yan/105293041

Clock.cs

```
namespace ClockProgram
{
    public class Clock
    {
        // Fields
        private Counter _hour;
        private Counter _minute;
        private Counter _second;

        // Constructor
        public Clock()
        {
            _hour = new Counter("Hour");
            _minute = new Counter("Minute");
            _second = new Counter("Second");
        }

        // Methods
        public void Tick()
        {
            this.IncrementSecond();
        }

        public void Reset()
        {
            _second.Reset();
            _minute.Reset();
            _hour.Reset();
        }

        private void IncrementSecond()
        {
            _second.Increment();
            if (_second.Ticks == 60)
            {
                _second.Reset();
                this.IncrementMinute();
            }
        }
        private void IncrementMinute()
        {
            _minute.Increment();
            if (_minute.Ticks == 60)
            {
                _minute.Reset();
                this.IncrementHour();
            }
        }
        private void IncrementHour()
        {
            _hour.Increment();
            if (_hour.Ticks == 13)
            {
                _hour.Reset();
                _hour.Increment();
            }
        }

        public string GetTime()
        {
            return $"{this.Hour}:{this.Minute}:{this.Second}";
        }

        // Properties
        private string Hour
        {
            get
```

```
            {
                if (_hour.Ticks == 0) _hour.Increment();
                return _hour.Ticks.ToString("D2");
            }
        }
        private string Minute
        {
            get { return _minute.Ticks.ToString("D2"); }
        }
        private string Second
        {
            get { return _second.Ticks.ToString("D2"); }
        }
    }
}
```

Program.cs

```
namespace ClockProgram
{
    public class Program
    {
        public static void Main(string[] args)
        {
            int secondsInADay = 86400;
            Clock myClock = new Clock();
            for (int i = 0; i < secondsInADay; i++)
            {
                myClock.Tick();
                Console.WriteLine(myClock.GetTime());
            }
        }
    }
}
```

TestCounter.cs

```
using NUnit.Framework;
using NUnit.Framework.Legacy;
using ClockProgram;
namespace CounterTest
{
    [TestFixture]
    public class CounterTest
    {
        private Counter _counter;
        [SetUp]
        public void Setup()
        {
            _counter = new Counter("Test");
        }

        [Test]
        public void TestInitialize()
        {
            // ClassicAssert.AreEqual(0, _counter.Ticks);
            Assert.That(_counter.Ticks, Is.EqualTo(0));
        }

        [Test]
        public void TestOneIncrement()
        {
            _counter.Increment();
            // ClassicAssert.AreEqual(1, _counter.Ticks);
            Assert.That(_counter.Ticks, Is.EqualTo(1));
        }

        [Test]
        public void TestNIncrement()
        {
            _counter = new Counter("Test"); // create a new obj for testing
            int nIncrement = 10;
            for (int i = 0; i < nIncrement; i++)
            {
                _counter.Increment();
            }

            // ClassicAssert.AreEqual(nIncrement, _counter.Ticks);
            Assert.That(_counter.Ticks, Is.EqualTo(nIncrement));
        }

        public void TestReset()
        {
            _counter.Reset();
```

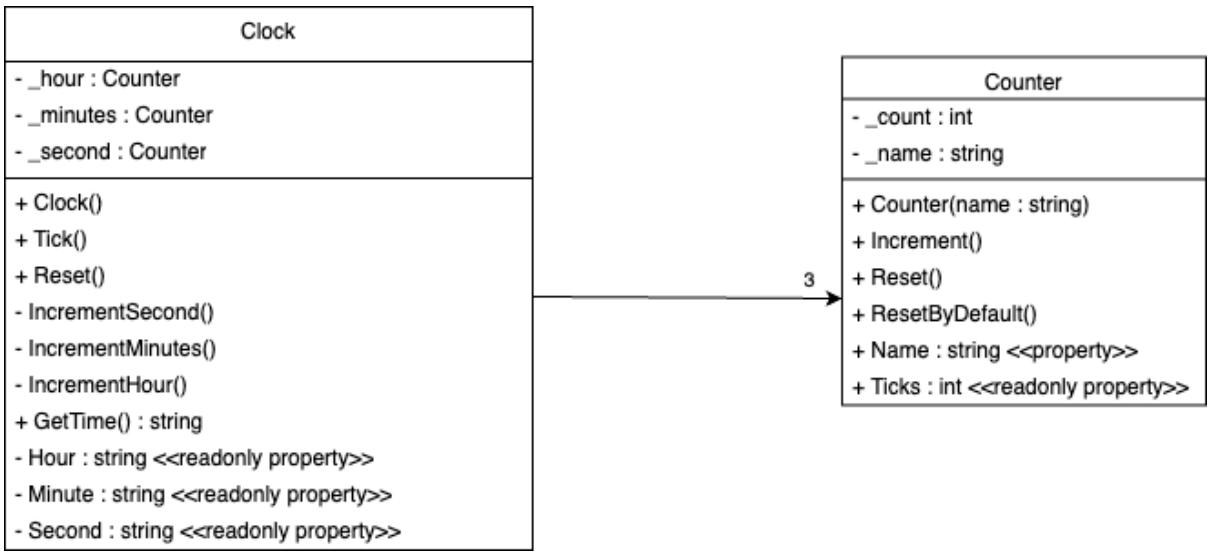
```
            Assert.That(_counter.Ticks, Is.EqualTo(0));
        }
    }
}
```

TestClock.cs

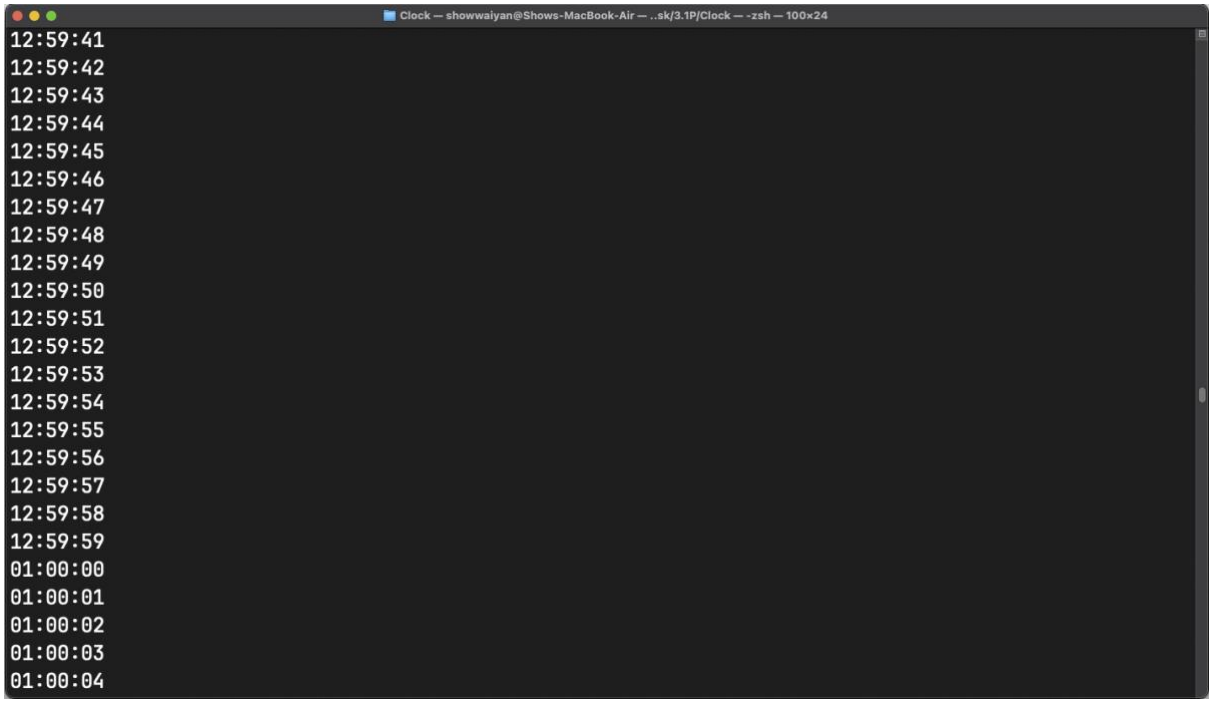
```
using NUnit.Framework;
using ClockProgram;

namespace ClockTest
{
    [TestFixture]
    public class Tests
    {
        private Clock clock;
        [SetUp]
        public void Setup()
        {
            clock = new Clock();
        }

        [Test]
        public void TestOneSecond()
        {
            clock.Reset();
            clock.Tick();
            Assert.That(clock.GetTime(), Is.EqualTo("01:00:01"));
        }

        [Test]
        public void TestNSecond()
        {
            clock.Reset();
            int second = 49;
            for (int i = 0; i < second; i++) clock.Tick();


            Assert.That(clock.GetTime(), Is.EqualTo($"01:00:{second.ToString("D2")}"));
        }

        [Test]
        public void TestOneMinute()
        {
            clock.Reset();
            int second = 60;
            for (int i = 0; i < second; i++) clock.Tick();
            Assert.That(clock.GetTime(), Is.EqualTo("01:01:00"));
        }

        [Test]
        public void TestNMinute()
        {
            clock.Reset();
            int minute = 3;
            int second = 60 * minute;
            for (int i = 0; i < second; i++) clock.Tick();
            Assert.That(clock.GetTime(), Is.EqualTo($"01:{minute.ToString("D2")}:00"));
        }

        [Test]
        public void TestSecondMinute()
        {
            clock.Reset();
            int second = 190;
            int minute = second / 60;
            for (int i = 0; i < second; i++) clock.Tick();
            Assert.That(clock.GetTime(), Is.EqualTo($"01:{minute.ToString("D2")}:{(second % 60).ToString("D2")}"));
        }

        [Test]
        public void TestOneHour()
        {
            clock.Reset();
            int second = 3600;
            for (int i = 0; i < second; i++) clock.Tick();
            Assert.That(clock.GetTime(), Is.EqualTo("01:00:00"));
        }

        [Test]
        public void TestNHour()
        {
            clock.Reset();
            int hour = 5;
            int second = 3600 * hour;
            for (int i = 0; i < second; i++) clock.Tick();
            Assert.That(clock.GetTime(), Is.EqualTo($"{hour.ToString("D2")}:00:00"));
        }

        [Test]
        public void TestHourMinuteSecond()
```

```
        {
            clock.Reset();
            int second = 5500;
            int minute = (second % 3600) / 60;
            int hour = second / 3600;
            for (int i = 0; i < second; i++) clock.Tick();
            Assert.That(clock.GetTime(), Is.EqualTo($"{hour.ToString("D2")}:{minute.ToString("D2")}:{(second %
60).ToString("D2")}"));
        }
    }
}
```

UML Class diagram

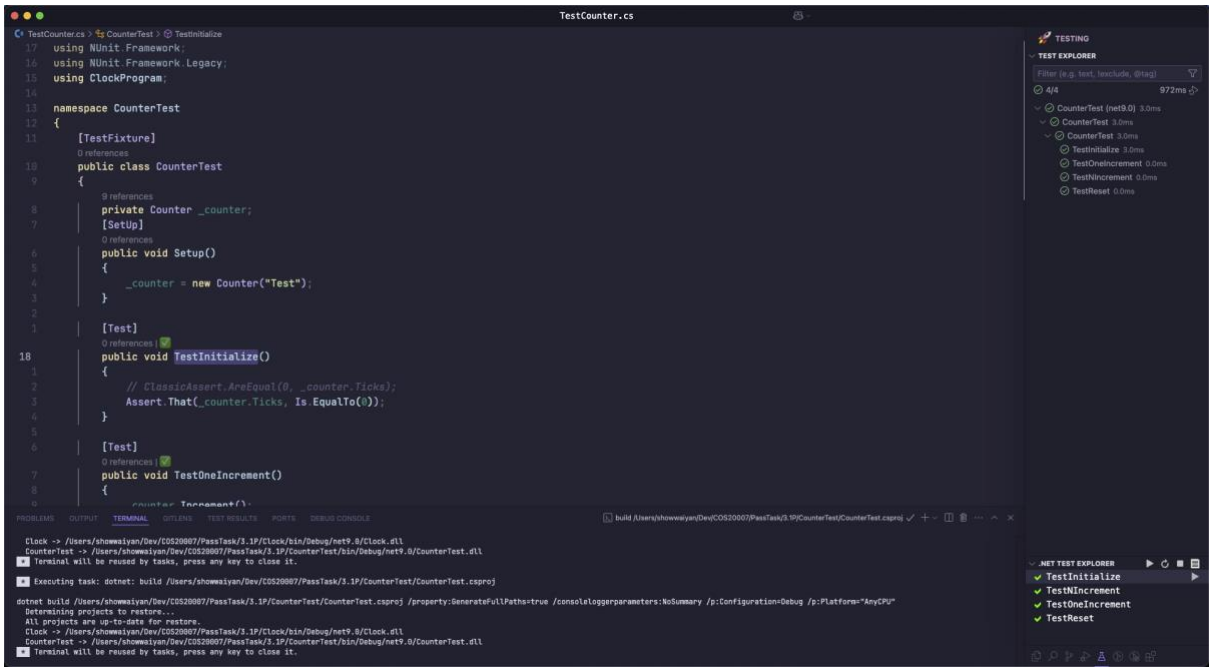


Screenshot of the console output



Screenshot of the Test Explorer showing your unit tests running

For TestCounter.cs



For TestClock.cs