```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6
 7  public class Clock
 8  {
 9      private Counter _hour;
10      private Counter _min;
11      private Counter _sec;
12
13      public Clock()
14      {
15          _hour = new Counter("Hour");
16          _min = new Counter("Minute");
17          _sec = new Counter("Second");
18      }
19
20      public void Tick()
21      {
22          if (_sec.Ticks <59)
23          {
24              _sec.Increment();
25          }
26          else
27          {
28              _sec.Reset();
29              if(_min.Ticks <59)
30              {
31                  _min.Increment();
32              }
33              else
34              {
35                  _min.Reset();
36                  if(_hour.Ticks <11)
37                  {
38                      _hour.Increment();
39                  }
40                  else
41                  {
42                      _hour.Reset();
43                  }
44              }
45          }
46      }
47
48      public void Reset()
49      {
```

```
50          _hour.Reset();
51          _min.Reset();
52          _sec.Reset();
53      }
54
55      public string ClockTime
56      {
57          get
58          {
59              return $"{_hour.Ticks:D2}:{_min.Ticks:D2}:{_sec.Ticks:D2}";
60          }
61      }
62  }
63
64
```

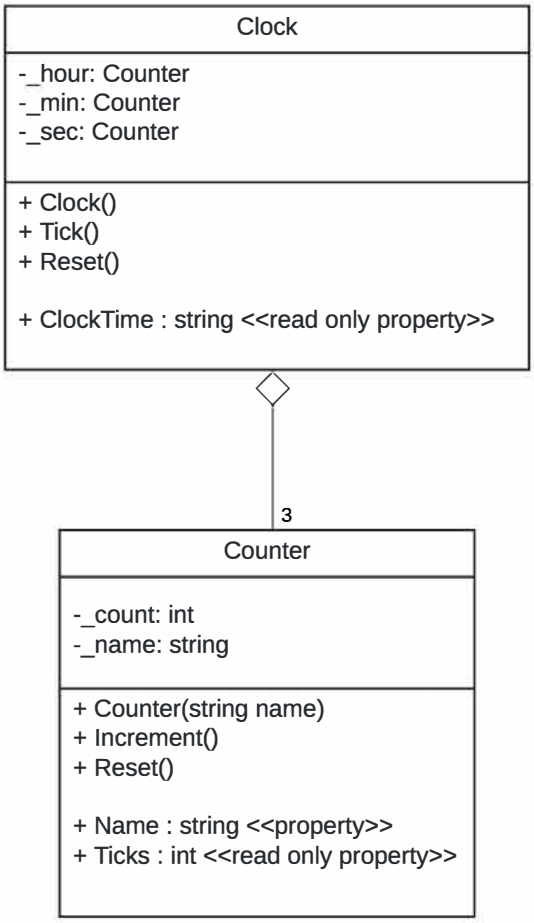
```
namespace ClockClassUnitTest
{
    public class Tests
    {
        Clock _clock_test;

        [SetUp]
        public void Setup()
        {
            _clock_test = new Clock();
        }

        [Test]
        public void TestClockStart()
        {
            Assert.AreEqual("00:00:00", _clock_test.ClockTime);
        }

        [Test]
        public void TestTick()
        {
            _clock_test.Tick();
            Assert.AreEqual("00:00:01", _clock_test.ClockTime);
        }

        [Test]
        public void TestReset()
        {
            _clock_test.Tick();
            _clock_test.Reset();
            Assert.AreEqual("00:00:00", _clock_test.ClockTime);
        }

        [TestCase(60)]
        public void TestFormat(int times)
        {
            for(int i=0; i<times; i++)
            {
                _clock_test.Tick();
            }
            Assert.AreEqual("00:01:00", _clock_test.ClockTime);
        }
    }
}
```

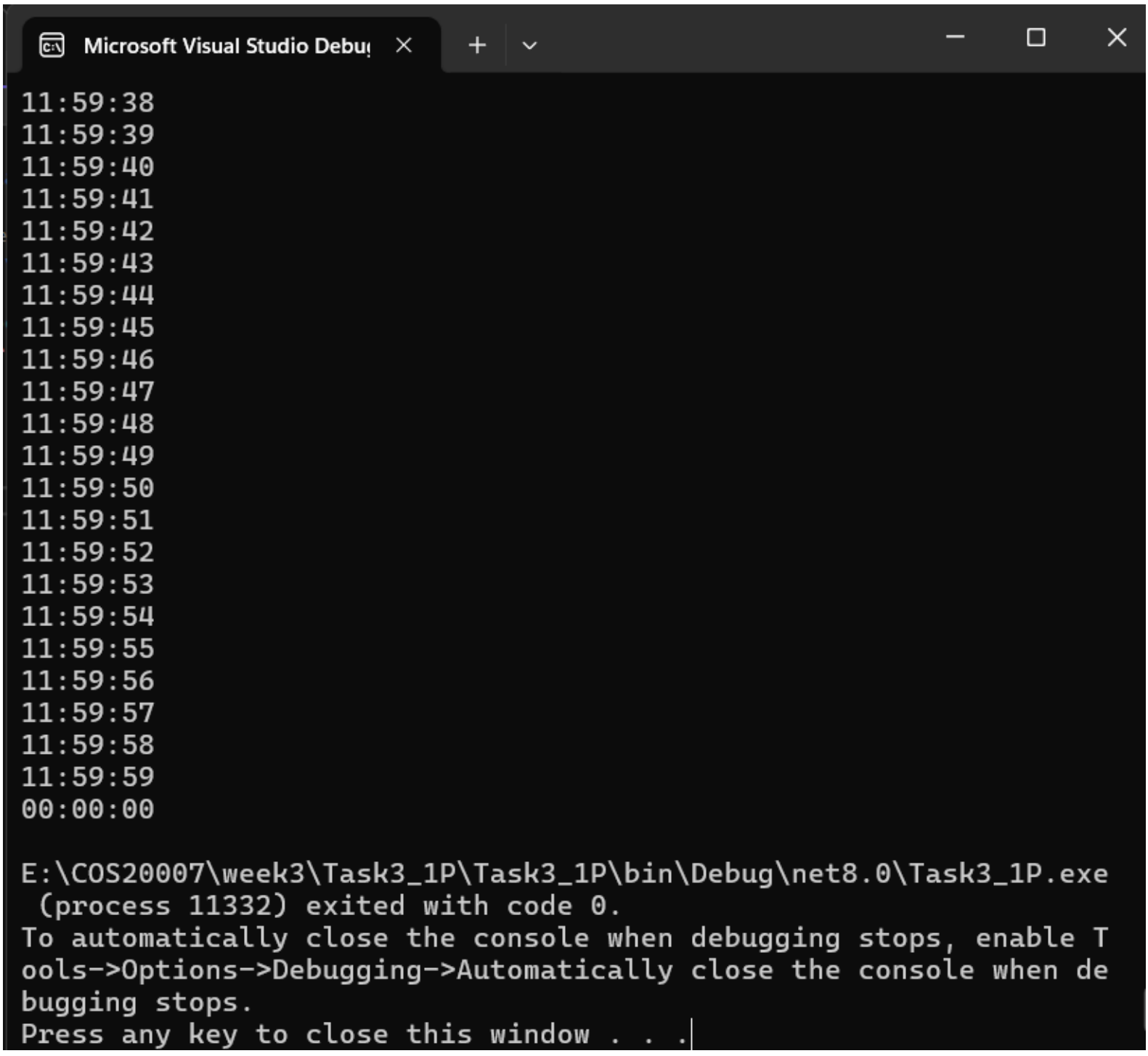
```
1  public class Counter
2  {
3      private int _count;
4      private string _name;
5
6      public Counter(string name)
7      {
8          _name = name;
9          _count = 0;
10     }
11
12     public void Increment()
13     {
14         _count++;
15     }
16
17     public void Reset()
18     {
19         _count = 0;
20     }
21
22     public string Name
23     {
24         get
25         {
26             return _name;
27         }
28         set
29         {
30             _name = value;
31         }
32     }
33
34     public int Ticks
35     {
36         get
37         {
38             return _count;
39         }
40     }
41 }
```

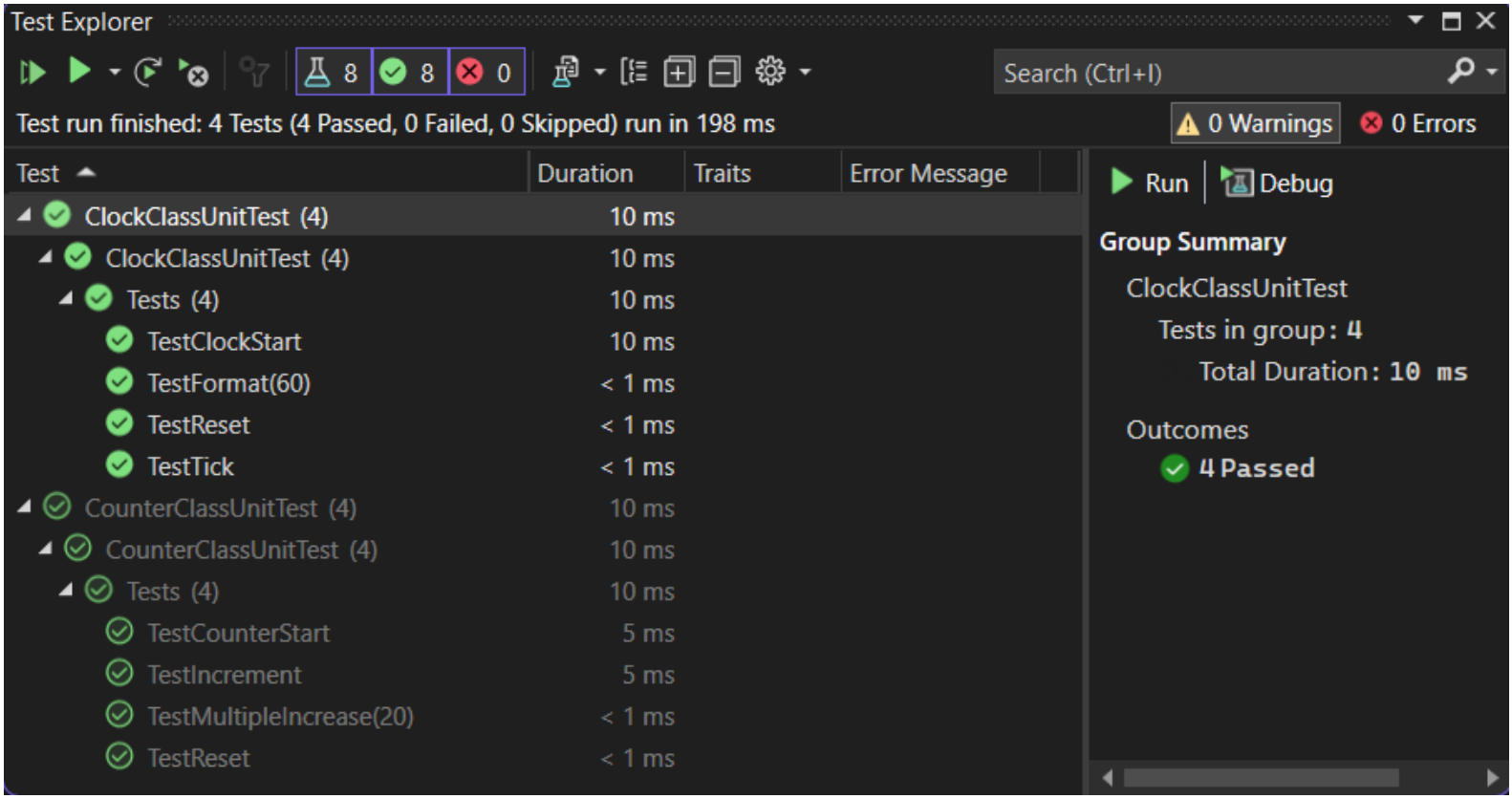
```
1  namespace CounterClassUnitTest
2  {
3      public class Tests
4      {
5          private Counter _cnt_test;
6          [SetUp]
7          public void Setup()
8          {
9              _cnt_test = new Counter("Test Counter");
10         }
11
12         [Test]
13         public void TestCounterStart()
14         {
15             Assert.IsTrue(_cnt_test.Ticks==0);
16         }
17
18         [Test]
19         public void TestIncrement()
20         {
21             _cnt_test.Increment();
22             Assert.AreEqual(_cnt_test.Ticks, 1);
23         }
24
25         [TestCase(20)]
26         public void TestMultipleIncrease(int count)
27         {
28             for(int i=0; i<count; i++)
29             {
30                 _cnt_test.Increment();
31             }
32             Assert.AreEqual(_cnt_test.Ticks, count);
33         }
34
35         [Test]
36         public void TestReset()
37         {
38             _cnt_test.Increment();
39             _cnt_test.Reset();
40             Assert.AreEqual(_cnt_test.Ticks, 0);
41         }
42     }
43 }
```

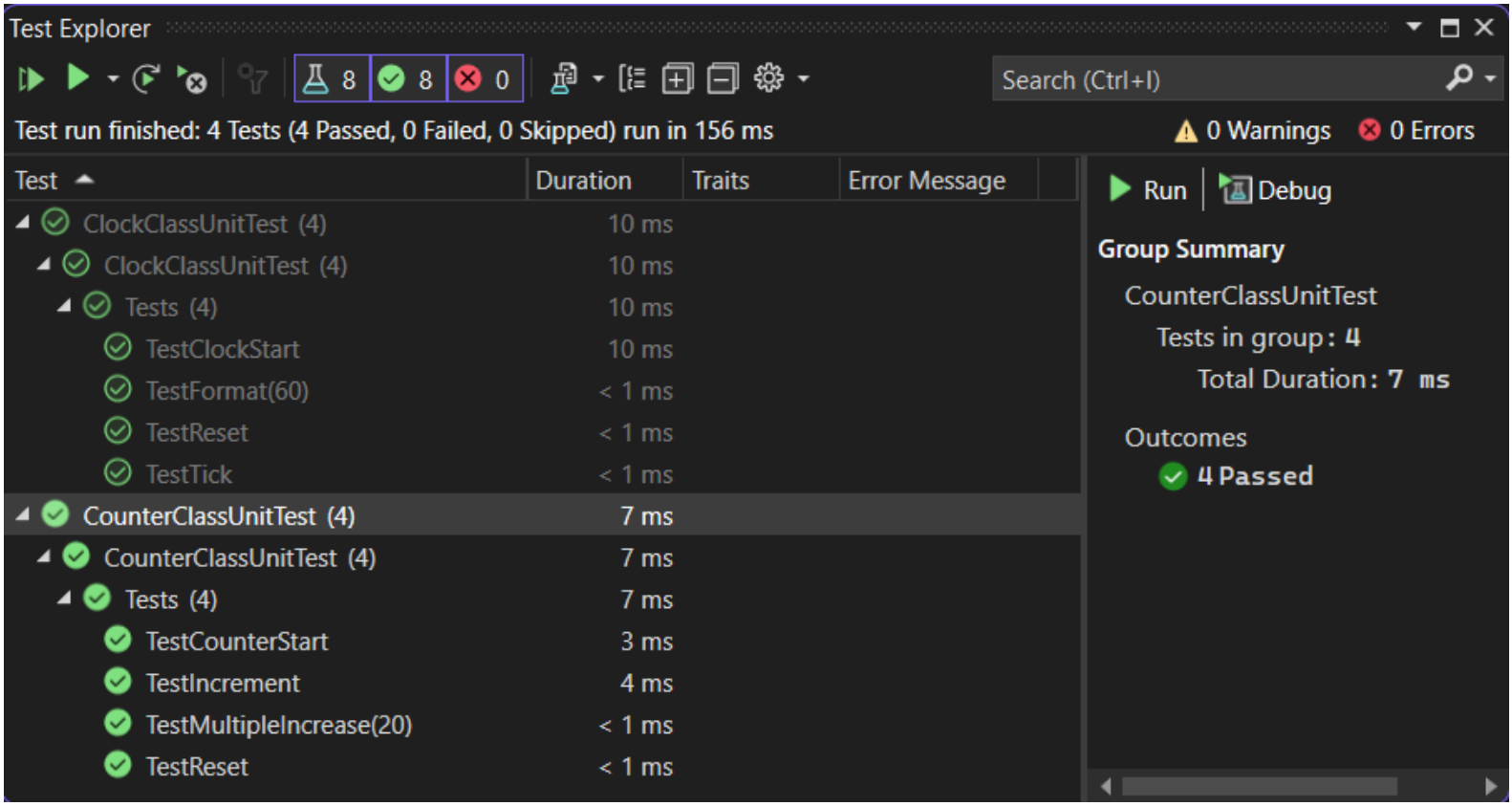
```
1  using System.Runtime.Serialization.Formatters;
2
3  namespace Task3_1P
4  {
5      internal class Program
6      {
7          static void Main(string[] args)
8          {
9              Clock myclock = new Clock();
10             for (int i=0; i<= 43200; i++) //43200
11             {
12                 Console.WriteLine(myclock.ClockTime);
13                 myclock.Tick();
14             }
15         }
16     }
17 }
18
```

UML diagram:


Clock
-_hour: Counter
-_min: Counter
-_sec: Counter
+ Clock()
+ Tick()
+ Reset()
+ ClockTime : string <<read only property>>
3
Counter
-_count: int
-_name: string
+ Counter(string name)
+ Increment()
+ Reset()
+ Name : string <<property>>
+ Ticks : int <<read only property>>


Microsoft Visual Studio Debu
11:59:38
11:59:39
11:59:40
11:59:41
11:59:42
11:59:43
11:59:44
11:59:45
11:59:46
11:59:47
11:59:48
11:59:49
11:59:50
11:59:51
11:59:52
11:59:53
11:59:54
11:59:55
11:59:56
11:59:57
11:59:58
11:59:59
00:00:00
E:\COS20007\week3\Task3_1P\Task3_1P\bin\Debug\net8.0\Task3_1P.exe
(process 11332) exited with code 0.
To automatically close the console when debugging stops, enable T
ools->Options->Debugging->Automatically close the console when de
bugging stops.
Press any key to close this window . . .


Test Explorer
8
8
0
Search (Ctrl+I)
Test run finished: 4 Tests (4 Passed, 0 Failed, 0 Skipped) run in 198 ms
0 Warnings
0 Errors
Test
Duration
Traits
Error Message
ClockClassUnitTest (4) 10 ms
ClockClassUnitTest (4) 10 ms
Tests (4) 10 ms
TestClockStart 10 ms
TestFormat(60) < 1 ms
TestReset < 1 ms
TestTick < 1 ms
CounterClassUnitTest (4) 10 ms
CounterClassUnitTest (4) 10 ms
Tests (4) 10 ms
TestCounterStart 5 ms
TestIncrement 5 ms
TestMultipleIncrease(20) < 1 ms
TestReset < 1 ms
Run
Debug
Group Summary
ClockClassUnitTest
Tests in group: 4
Total Duration: 10 ms
Outcomes
4 Passed


Test Explorer
8
8
0
Search (Ctrl+I)
Test run finished: 4 Tests (4 Passed, 0 Failed, 0 Skipped) run in 156 ms
0 Warnings
0 Errors
Test
Duration
Traits
Error Message
ClockClassUnitTest (4) 10 ms
ClockClassUnitTest (4) 10 ms
Tests (4) 10 ms
TestClockStart 10 ms
TestFormat(60) < 1 ms
TestReset < 1 ms
TestTick < 1 ms
CounterClassUnitTest (4) 7 ms
CounterClassUnitTest (4) 7 ms
Tests (4) 7 ms
TestCounterStart 3 ms
TestIncrement 4 ms
TestMultipleIncrease(20) < 1 ms
TestReset < 1 ms
Run
Debug
Group Summary
CounterClassUnitTest
Tests in group: 4
Total Duration: 7 ms
Outcomes
4 Passed