```csharp
1
   using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace _2._4P
8  {
9      public class IdentifiableObject
10     {
11
12         private List<string> _identifiers = new List<string>() ;
13         public IdentifiableObject(string[] identifiers)
14         {
15
16
17
18             foreach (string id in identifiers)
19             {
20                 AddIdentifier(id);
21             }
22         }
23
24         public void AddIdentifier(string identifier)
25         {
26             _identifiers.Add(identifier.ToLower());
27         }
28
29         public bool AreYou(string identifier)
30             { return _identifiers.Contains(identifier.ToLower()); }
31
32         public string FirstId
33         {
34             get
35             {
36                 if (_identifiers.Count > 0)
37                 {
38                     return _identifiers[0];
39                 }
40                 else
41                 {
42                     return "";
43                 }
44             }
45         }
46
47
48
49         public void PrivilegeEscalation(string pin)
50         {
51             if (pin == "2183" && _identifiers.Count > 0)
52             {
53                 _identifiers[0] = "7";
54             }
```

```
        }
...
54          }
55    }
56
```


Test Explorer

7 | 7 | 0

0 Warnings | 0 Errors

Test	Duration	Traits	Error Message
IdentifiableObjectTest (7)	24 ms		
NUnitTests (7)	24 ms		
TestIdentifiableObject (7)	24 ms		
TestAddId()	23 ms		
TestAreYou()	< 1 ms		
TestCaseSensitivity()	< 1 ms		
TestFirstId()	1 ms		
TestFirstIdWithNoId()	< 1 ms		
TestNotAreYou()	< 1 ms		
TestPrivilegeEscalation()	< 1 ms		

Run | Debug

Group Summary

IdentifiableObjectTest
Tests in group: 7
Total Duration: 24 ms

Outcomes
7 Passed

```csharp
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Threading.Tasks;
 5  using _2._4P;
 6  using NUnit.Framework;
 7
 8
 9  namespace NUnitTests
10  {
11      [TestFixture]
12      public class TestIdentifiableObject
13      {
14          private IdentifiableObject _objectIdentifier;
15          private IdentifiableObject _emptyObjectIdentifier;
16          private IdentifiableObject _objectIdentifierNoStuId;
17
18          [SetUp]
19          public void SetUp()
20          {
21              _objectIdentifier = new IdentifiableObject(new string[]
                    { "104772183", "Khoi Nguyen", "Pham" });
22              _emptyObjectIdentifier = new IdentifiableObject(new string[]
                     { });
23              _objectIdentifierNoStuId = new IdentifiableObject(new string
                    [] { "1111", "Khoi Nguyen", "Pham" });
24          }
25
26          [TestCase]
27          public void TestAreYou()
28          {
29              Assert.IsTrue(_objectIdentifier.AreYou("104772183"));
30              Assert.IsTrue(_objectIdentifier.AreYou("Khoi Nguyen"));
31          }
32
33          [TestCase]
34          public void TestNotAreYou()
35          {
36              Assert.IsFalse(_objectIdentifier.AreYou("nonexistent
                    identifier"));
37          }
38
39
40
41          [TestCase]
42          public void TestFirstId()
43          {
44              Assert.AreEqual("104772183", _objectIdentifier.FirstId);
45          }
46
47          [TestCase]
48          public void TestFirstIdWithNoId()
49          {
```

```csharp
50              Assert.AreEqual("", _emptyObjectIdentifier.FirstId);
51          }
52
53          [TestCase]
54          public void TestAddId()
55          {
56              _objectIdentifier.AddIdentifier("Nevan");
57              Assert.IsTrue(_objectIdentifier.AreYou("Nevan"));
58          }
59
60          [TestCase]
61          public void TestCaseSensitivity()
62          {
63              Assert.IsTrue(_objectIdentifier.AreYou("khOI NGUyEN"));
64              Assert.IsTrue(_objectIdentifier.AreYou("PhAM"));
65          }
66
67          [TestCase]
68          public void TestPrivilegeEscalation()
69          {
70
71
72              string correctPin = "2183";
73              string expectedFirstId = "7";
74
75
76              _objectIdentifierNoStuId.PrivilegeEscalation(correctPin);
77
78              // Assert
79              Assert.AreEqual(expectedFirstId,
80                _objectIdentifierNoStuId.FirstId);
81          }
82      }
83
84
85
86  }
87
88
```