COS20007 - Object Oriented Programming

6.1P - Case Study - Iteration 4: Look Command with customization

Student name: Nguyen Duc Manh
ID: 105547489

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public abstract class Command : IdenObj
10     {
11         public Command(string[] ids) : base(ids)
12         {
13             //
14         }
15
16         public abstract string Execute(Player p, string[] text);
17     }
18 }
19
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7  using static System.Runtime.InteropServices.JavaScript.JSType;
8  using System.Xml.Linq;
9
10 namespace SwinAdventure
11 {
12     public class LookCommand : Command
13     {
14         IHaveInventory container;
15         GameObject item;
16         Player p;
17
18         public LookCommand() : base(["look"]) { }
19
20         public override string Execute(Player p, string[] text)
21         {
22             if (text.Length == 3 || text.Length == 5)
23             {
24                 if (text[0] != "look")
25                     return "Error in look input";
26                 if (text[1] != "at")
27                     return "What do you want to look at?";
28                 if (text.Length == 5 && text[3] != "in")
29                     return "What do you want to look in?";
30                 if (text.Length == 3)
31                 {
32                     container = p;
33                 }
34                 else
35                 {
36                     container = FetchContainer(p, text[4]);
37                     if (container == null)
38                         return $"I cannot find the {text[4]}";
39                 }
40
41                 return LookAtIn(text[2], container);
42             }
43             else
44                 return "I don't know how to look like that";
45         }
46
47         private IHaveInventory? FetchContainer(Player p, string
           containerId)
48         {
```

```
49              return p.Locate(containerId) as IHaveInventory;
50          }
51
52          private string LookAtIn(string thingId, IHaveInventory container)
53          {
54              if (container.Locate(thingId) != null)
55              {
56                  return container.Locate(thingId).FullDescription;
57              }
58              else
59                  return $"I cannot find the {thingId}";
60          }
61      }
62  }
63
```

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6
 7  namespace SwinAdventure
 8  {
 9      public interface IHaveInventory
10      {
11          public GameObject Locate(string id);
12
13          public string Name { get; }
14      }
15  }
16
```

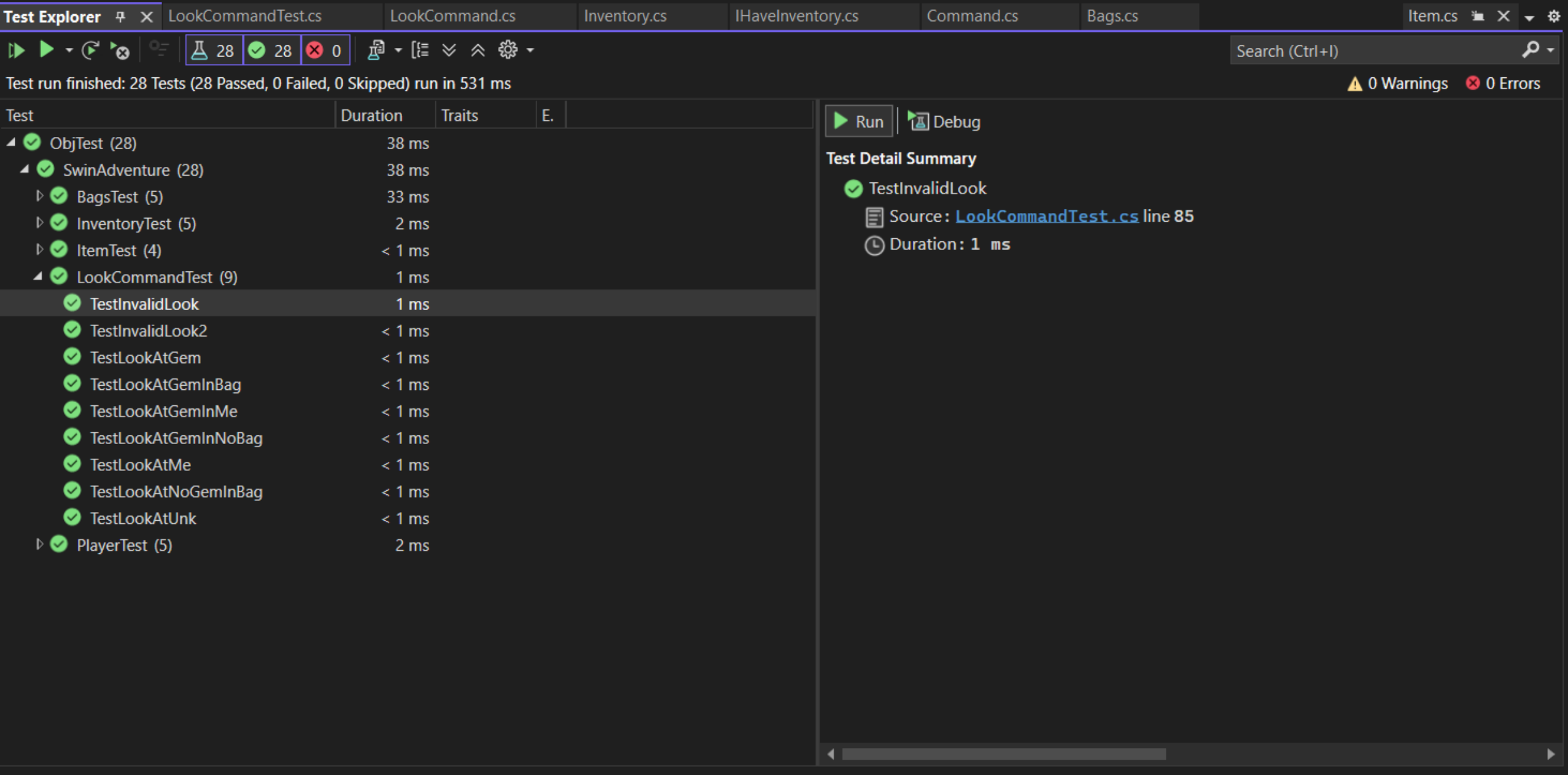
```
1  using System.Numerics;
2
3  namespace SwinAdventure;
4
5  public class LookCommandTest
6  {
7      Player me;
8      Item gem;
9      Bags bag;
10     Command look;
11
12     [SetUp]
13     public void Setup()
14     {
15         me = new Player("me", "the main character of the game");
16         gem = new Item(["gem"], "a bright red stone", "it sparkles in the
             light");
17         bag = new Bags(["bag"], "a small bag", "it has a zipper");
18         look = new LookCommand();
19
20         me.Inventory.Put(gem);
21         me.Inventory.Put(bag);
22         bag.Inventory.Put(gem);
23     }
24
25     [Test]
26     public void TestLookAtMe()
27     {
28         string prompt = look.Execute(me, ["look", "at", "inventory"]);
29         string expected = me.FullDescription;
30         Assert.That(prompt, Is.EqualTo(expected));
31     }
32
33     [Test]
34     public void TestLookAtGem()
35     {
36         string prompt = look.Execute(me, ["look", "at", "gem"]);
37         string expected = gem.FullDescription;
38         Assert.That(prompt, Is.EqualTo(expected));
39     }
40
41     [Test]
42     public void TestLookAtUnk()
43     {
44         string prompt = look.Execute(me, ["look", "at", "unk"]);
45         string expected = "I cannot find the unk";
46         Assert.That(prompt, Is.EqualTo(expected));
47     }
48
```

```
49      [Test]
50      public void TestLookAtGemInMe()
51      {
52          string prompt = look.Execute(me, ["look", "at", "gem", "in",
              "inventory"]);
53          string expected = gem.FullDescription;
54          Assert.That(prompt, Is.EqualTo(expected));
55      }
56
57      [Test]
58      public void TestLookAtGemInBag()
59      {
60          Assert.That(me.Locate("bag"), Is.EqualTo(bag)); //test that bag
              is in player's inventory
61          string prompt = look.Execute(me, ["look", "at", "gem", "in",
              "bag"]);
62          string expected = gem.FullDescription;
63          Assert.That(prompt, Is.EqualTo(expected));
64      }
65
66      [Test]
67      public void TestLookAtGemInNoBag()
68      {
69          me.Inventory.Take("bag"); //remove bag from player's inventory
70          string prompt = look.Execute(me, ["look", "at", "gem", "in",
              "bag"]);
71          string expected = "I cannot find the bag";
72          Assert.That(prompt, Is.EqualTo(expected));
73      }
74
75      [Test]
76      public void TestLookAtNoGemInBag()
77      {
78          bag.Inventory.Take("gem"); //remove gem from player's inventory
79          string prompt = look.Execute(me, ["look", "at", "gem", "in",
              "bag"]);
80          string expected = "I cannot find the gem";
81          Assert.That(prompt, Is.EqualTo(expected));
82      }
83
84      [Test]
85      public void TestInvalidLook()
86      {
87          string prompt = look.Execute(me, ["look", "at", "gem", "not in",
              "bag"]);
88          string expected = "What do you want to look in?";
89          Assert.That(prompt, Is.EqualTo(expected));
90      }
91
```

```
92      [Test]
93      public void TestInvalidLook2()
94      {
95          string prompt = look.Execute(me, ["kool", "at", "gem", "in",
              "bag"]);
96          string expected = "Error in look input";
97          Assert.That(prompt, Is.EqualTo(expected));
98      }
99  }
100
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public class Player : GameObject, IHaveInventory
10     {
11         Inventory _inventory;
12
13         public Player (string name, string desc) : base(new string[] {"
             me", "inventory"}, name, desc)
14         {
15             _inventory = new Inventory();
16         }
17
18         public GameObject Locate(string id)
19         {
20             if (AreYou(id))
21             {
22                 return this;
23             }
24             return _inventory.Fetch(id);
25         }
26
27         public override string FullDescription
28         {
29             get
30             {
31                 return $"{Name}, {base.ShortDescription}.You are carrying:
                     {_inventory.ItemList()}";
32             }
33         }
34
35         public Inventory Inventory { get { return _inventory; } }
36     }
37 }
38
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public class Bags : Item, IHaveInventory
10     {
11         Inventory _inventory;
12
13         public Bags(string[] idents, string name, string desc) : base
             (idents, name, desc)
14         {
15             _inventory = new Inventory();
16         }
17
18         public GameObject Locate(string id)
19         {
20             if (AreYou(id))
21             {
22                 return this;
23             }
24             else if (_inventory.HasItem(id))
25             {
26                 return _inventory.Fetch(id);
27             } return null;
28         }
29
30         public override string FullDescription
31         {
32             get { return $"In the {Name} you can see:\n{_inventory.ItemList
                 ()}" ; }
33         }
34
35         public Inventory Inventory
36         {
37             get { return _inventory; }
38         }
39     }
40 }
41
```


Test Explorer
LookCommandTest.cs
LookCommand.cs
Inventory.cs
IHaveInventory.cs
Command.cs
Bags.cs
Item.cs
28
28
0
Search (Ctrl+I)
Test run finished: 28 Tests (28 Passed, 0 Failed, 0 Skipped) run in 531 ms
0 Warnings
0 Errors
Test
Duration
Traits
E.
ObjTest (28) 38 ms
SwinAdventure (28) 38 ms
BagsTest (5) 33 ms
InventoryTest (5) 2 ms
ItemTest (4) < 1 ms
LookCommandTest (9) 1 ms
TestInvalidLook 1 ms
TestInvalidLook2 < 1 ms
TestLookAtGem < 1 ms
TestLookAtGemInBag < 1 ms
TestLookAtGemInMe < 1 ms
TestLookAtGemInNoBag < 1 ms
TestLookAtMe < 1 ms
TestLookAtNoGemInBag < 1 ms
TestLookAtUnk < 1 ms
PlayerTest (5) 2 ms
Run
Debug
Test Detail Summary
TestInvalidLook
Source: LookCommandTest.cs line 85
Duration: 1 ms

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public class Inventory
10     {
11         List<Item> _items;
12
13         public Inventory()
14         {
15             _items = new List<Item>();
16         }
17
18         public bool HasItem(string id)
19         {
20             foreach (Item item in _items)
21             {
22                 if (item.AreYou(id))
23                 {
24                     return true;
25                 }
26             }
27             return false;
28         }
29
30         public void Put(Item itm)
31         {
32             _items.Add(itm);
33         }
34
35         //public void RemoveItm(Item itm)
36         //{
37         //    if (_items.Contains(itm))
38         //    {
39         //        _items.Remove(itm);
40         //    }
41         //}
42
43         public Item Take(string id)
44         {
45             foreach (Item item in _items)
46             {
47                 if (item.AreYou(id))
48                 {
49                     _items.Remove(item);
```

```
50                      return item;
51                  }
52              }
53              return null;
54          }
55
56          public Item Fetch(string id)
57          {
58              foreach (Item item in _items)
59              {
60                  if (item.AreYou(id))
61                  {
62                      return item;
63                  }
64              }
65              return null;
66          }
67
68          public string ItemList()
69          {
70              string listitm = "";
71              foreach (Item item in _items)
72              {
73                  listitm = listitm + item.ShortDescription + "\n";
74              }
75              return listitm;
76          }
77      }
78  }
79
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public abstract class GameObject : IdenObj
10     {
11         string _description;
12         string _name;
13
14         public GameObject(string[] idents, string name, string desc) : base
             (idents)
15         {
16             _name = name;
17             _description = desc;
18         }
19
20         public string Name { get { return _name; } }
21
22         public string ShortDescription { get { return $"{_name}
             ({FirstId})"; } }
23
24         public virtual string FullDescription { get { return
             _description; } }
25     }
26 }
27
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public class IdenObj
10     {
11         //fields
12         private List<string> _identifiers;
13         string _myStudentID = "7489";
14
15         //constructor
16         public IdenObj(string[] idents)
17         {
18             _identifiers = new List<string>();
19             if (idents != null)
20             {
21                 for (int i = 0; i < idents.Length; i++)
22                 {
23                     _identifiers.Add(idents[i].ToLower());
24                 }
25             }
26         }
27
28         //methods
29         public bool AreYou(string id)
30         {
31             return _identifiers.Contains(id.ToLower());
32         }
33
34         public string FirstId
35         {
36             get
37             {
38                 if( _identifiers.Count == 0)
39                 {
40                     return "";
41                 } else { return _identifiers.First(); }
42             }
43         }
44
45         public void AddIdentifier(string id)
46         {
47             _identifiers.Add(id.ToLower());
48         }
49
```

```
50          public void PrivilegeEscalation(string pin)
51          {
52              if(pin.Length == 4)
53              {
54                  if(pin == _myStudentID) //105547489
55                  {
56                      _identifiers[0] = _myStudentID;
57                  }
58              }
59              else
60              {
61                  return;
62              }
63
64          }
65      }
66  }
67
```

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6
 7  namespace SwinAdventure
 8  {
 9      public class Item : GameObject
10      {
11          public Item(string[] idents, string name, string desc) : base
              (idents, name, desc)
12          {
13              //not yet
14          }
15      }
16  }
17
```