

SWINBURNE UNIVERSITY OF TECHNOLOGY

COS20007 OBJECT ORIENTED PROGRAMMING

7.2C - Case Study - Iteration 6 - Locations

PDF generated at 15:26 on Monday 15th May, 2023

```
1  using System;
2  using System.IO;
3
4  namespace SwinAdventure
5  {
6      public class Location : GameObject, IHaveInventory
7      {
8          private Inventory _inventory;
9
10         public Location(string[] ids, string name, string desc) : base(ids, name,
↪ desc)
11         {
12             _inventory = new Inventory();
13
14             AddIdentifier("room");
15             AddIdentifier("here");
16         }
17
18         public GameObject Locate(string id)
19         {
20             if (AreYou(id))
21             {
22                 return this;
23             }
24
25             return _inventory.Fetch(id);
26         }
27
28         // Locations will need to be identifiable and have a name, and description.
29         public override string FullDescription
30         {
31             get
32             {
33                 return $"You are in {Name}\n{base.FullDescription}\nIn this room you
↪ can see:\n{Inventory.ItemList}";
34             }
35         }
36
37         // Location can contain items
38         public Inventory Inventory
39         {
40             get
41             {
42                 return _inventory;
43             }
44         }
45     }
46 }
```

```
1 using System.Security.Principal;
2
3 namespace SwinAdventure
4 {
5     public class LocationTest
6     {
7         private Player _testPlayer;
8         private Location _testLocation;
9
10        private Item _sword;
11        private Item _shovel;
12        private Item _pc;
13
14        [SetUp]
15        public void SetUp()
16        {
17            _testPlayer = new Player("Trung Kien Nguyen", "I am the player");
18            _testLocation = new Location(new string[] { "studio" }, "a studio", "A
↵ small, beautiful and fully-furnished studio.");
19
20            _sword = new Item(new string[] { "sword", "bronze" }, "a bronze sword",
↵ "This is a bronze sword");
21            _shovel = new Item(new string[] { "shovel" }, "a shovel", "This is a
↵ shovel");
22            _pc = new Item(new string[] { "pc", "computer" }, "a small computer",
↵ "This is a small computer");
23
24            _testPlayer.Inventory.Put(_sword);
25
26            _testLocation.Inventory.Put(_shovel);
27            _testLocation.Inventory.Put(_pc);
28        }
29
30        // Locations can identify themselves
31        [Test]
32        public void TestLocationLocateItself()
33        {
34            Assert.AreEqual(_testLocation.Locate("room"), _testLocation);
35            Assert.AreEqual(_testLocation.Locate("here"), _testLocation);
36            Assert.AreEqual(_testLocation.Locate("studio"), _testLocation);
37
38        }
39
40        // Locations can locate items they have
41        [Test]
42        public void TestLocationLocateItems()
43        {
44            // Locate items that are in the location inventory
45            Assert.AreEqual(_testLocation.Locate("shovel"), _shovel);
46            Assert.AreEqual(_testLocation.Locate("pc"), _pc);
47
48            // Locate item that is not in the location inventory
49            Assert.AreEqual(_testLocation.Locate("sword"), null);
```

```
50     }
51
52     [Test]
53     public void TestLocationFullDescription()
54     {
55         Assert.AreEqual(_testLocation.FullDescription, $"You are in
↪ {_testLocation.Name}\nA small, beautiful and fully-furnished studio.\nIn this
↪ room you can see:\n{_testLocation.Inventory.ItemList}");
56     }
57 }
58 }
```

```

1  using System;
2  using System.Net.NetworkInformation;
3  using System.Text.RegularExpressions;
4
5  namespace SwinAdventure
6  {
7      public class Player : GameObject, IHaveInventory
8      {
9          private Inventory _inventory;
10         private Location _location;
11
12         public Player(string name, string desc) : base( new string[] { "me",
↪ "inventory"}, name, desc)
13         {
14             _inventory = new Inventory();
15         }
16
17         // Players "locate" items by checking three things (in order):
18         public GameObject Locate(string id)
19         {
20             // First checking if they are what is to be located (locate inventory)
21             if (AreYou(id))
22             {
23                 return this;
24             }
25
26             // Second, checking if they have what is being located (_inventory fetch
↪ gem)
27             if (_inventory.Fetch(id) != null)
28             {
29                 return _inventory.Fetch(id);
30             }
31
32             // Lastly, checking if the item can be located where they are (
↪ _location, locate gem)
33             if (_location != null)
34             {
35                 return _location.Locate(id);
36             }
37
38             return null;
39         }
40
41         public override string FullDescription
42         {
43             get
44             {
45                 return $"You are {Name}, ({base.FullDescription}), you are
↪ carrying:\n" + _inventory.ItemList;
46             }
47         }
48
49         public Inventory Inventory

```

```
50     {
51         get
52         {
53             return _inventory;
54         }
55     }
56
57     // Players have a location.
58     public Location Location
59     {
60         get
61         {
62             return _location;
63         }
64         set
65         {
66             _location = value;
67         }
68     }
69 }
70 }
```

```
1 namespace SwinAdventure
2 {
3     public class PlayerTest
4     {
5         private Player _testPlayer;
6         private Location _testLocation;
7         private Item _sword;
8         private Item _shovel;
9         private Item _pc;
10
11         [SetUp]
12         public void SetUp()
13         {
14             _testPlayer = new Player("Trung Kien Nguyen", "I am the player");
15             _testLocation = new Location(new string[] { "studio" }, "a studio", "A
↵ small, beautiful and fully-furnished studio.");
16
17             _sword = new Item(new string[] { "sword", "bronze" }, "a bronze sword",
↵ "This is a bronze sword");
18             _shovel = new Item(new string[] { "shovel" }, "a shovel", "This is a
↵ shovel");
19             _pc = new Item(new string[] { "pc", "computer" }, "a small computer",
↵ "This is a small computer");
20         }
21
22         [Test]
23         public void TestPlayerIsIdentifiable()
24         {
25             _testPlayer.Inventory.Put(_sword);
26             _testPlayer.Inventory.Put(_shovel);
27             _testPlayer.Inventory.Put(_pc);
28
29             Assert.IsTrue(_testPlayer.AreYou("me"));
30             Assert.IsTrue(_testPlayer.AreYou("inventory"));
31         }
32
33         [Test]
34         public void TestPlayerLocatesItems()
35         {
36             _testPlayer.Inventory.Put(_sword);
37             _testPlayer.Inventory.Put(_shovel);
38             _testPlayer.Inventory.Put(_pc);
39
40             GameObject locatedItem1 = _testPlayer.Locate("shovel");
41             GameObject locatedItem2 = _testPlayer.Locate("pc");
42
43             // Test if player has the located item
44             Assert.AreEqual(locatedItem1, _shovel);
45             Assert.AreEqual(locatedItem2, _pc);
46
47             // Test if the item remains in the player's inventory
48             Assert.IsTrue(_testPlayer.Inventory.HasItem("shovel"));
49             Assert.IsTrue(_testPlayer.Inventory.HasItem("pc"));
```

```
50     }
51
52     [Test]
53     public void TestPlayerLocatesItself()
54     {
55         _testPlayer.Inventory.Put(_sword);
56         _testPlayer.Inventory.Put(_shovel);
57         _testPlayer.Inventory.Put(_pc);
58
59         GameObject playerItself1 = _testPlayer.Locate("me");
60         GameObject playerItself2 = _testPlayer.Locate("inventory");
61
62         // Test if player has the located itself with the keyword "me"
63         Assert.AreEqual(playerItself1, _testPlayer);
64
65         // Test if player has the located itself with the keyword "inventory"
66         Assert.AreEqual(playerItself2, _testPlayer);
67     }
68
69     [Test]
70     public void TestPlayerLocatesNothing()
71     {
72         _testPlayer.Inventory.Put(_sword);
73         _testPlayer.Inventory.Put(_shovel);
74         _testPlayer.Inventory.Put(_pc);
75
76         GameObject nonExistentObject = _testPlayer.Locate("gun");
77
78         Assert.AreEqual(nonExistentObject, null);
79     }
80
81     [Test]
82     public void TestPlayerFullDescription()
83     {
84         _testPlayer.Inventory.Put(_sword);
85         _testPlayer.Inventory.Put(_shovel);
86         _testPlayer.Inventory.Put(_pc);
87
88         string playerFullDesc = $"You are {_testPlayer.Name}, (I am the player),
↪ you are carrying:\n{_testPlayer.Inventory.ItemList}";
89
90         Assert.AreEqual(playerFullDesc, _testPlayer.FullDescription);
91     }
92
93     [Test]
94     public void TestPlayersCanLocateTheirLocation()
95     {
96         _testPlayer.Inventory.Put(_sword);
97
98         _testLocation.Inventory.Put(_shovel);
99         _testLocation.Inventory.Put(_pc);
100
101         _testPlayer.Location = _testLocation;
```



```
102
103     Assert.AreEqual(_testPlayer.Locate("room"), _testLocation);
104     Assert.AreEqual(_testPlayer.Locate("here"), _testLocation);
105 }
106
107 // Players can locate items in their location
108 [Test]
109 public void TestPlayerCanLocateItemsInTheirLocation()
110 {
111     _testPlayer.Inventory.Put(_sword);
112
113     _testLocation.Inventory.Put(_shovel);
114     _testLocation.Inventory.Put(_pc);
115
116     _testPlayer.Location = _testLocation;
117
118     // Locate items that are in the player location inventory, but not in
↪ the player inventory
119     Assert.AreEqual(_testPlayer.Locate("shovel"), _shovel);
120     Assert.AreEqual(_testPlayer.Locate("pc"), _pc);
121
122     Assert.IsFalse(_testPlayer.Inventory.HasItem("shovel"));
123     Assert.IsFalse(_testPlayer.Inventory.HasItem("pc"));
124 }
125 }
126 }
```

```
1  using System;
2
3  namespace SwinAdventure
4  {
5      public class LookCommand : Command
6      {
7          public LookCommand() : base(new string[] { "look" })
8          {
9          }
10
11         public override string Execute(Player p, string[] text)
12         {
13             if ((text.Length == 1) || (text.Length == 3) || (text.Length == 5))
14             {
15                 if (text[0].ToLower() != "look")
16                 {
17                     return "Error in look input";
18                 }
19
20                 IHaveInventory container;
21                 string itemId;
22
23                 // This will change the look command to also include "look" to look
24                 ↪ at the player's location.
25                 if (text.Length == 1)
26                 {
27                     container = p as IHaveInventory;
28                     itemId = "room";
29                 }
30                 else
31                 {
32                     if (text[1].ToLower() != "at")
33                     {
34                         return "What do you want to look at?";
35                     }
36
37                     if (text.Length == 3)
38                     {
39                         container = p as IHaveInventory;
40                         itemId = text[2];
41                     }
42                     else
43                     {
44                         if (text[3].ToLower() != "in")
45                         {
46                             return "What do you want to look in?";
47                         }
48
49                         container = FetchContainer(p, text[4]);
50                         if (container == null)
51                         {
52                             return $"I cannot find the {text[4]}";
53                         }
54                     }
55                 }
56             }
57         }
58     }
59 }
```

```
53         itemId = text[2];
54     }
55 }
56
57     return LookAtIn(itemId, container);
58 }
59
60     return "I don't know how to look like that";
61 }
62
63 private IHaveInventory FetchContainer(Player p, string containerId)
64 {
65     return p.Locate(containerId) as IHaveInventory;
66 }
67
68 private string LookAtIn(string thingId, IHaveInventory container)
69 {
70     if (container.Locate(thingId) != null)
71     {
72         return container.Locate(thingId).FullDescription;
73     }
74
75     return $"I cannot find the {thingId} in the {container.Name}";
76 }
77 }
78 }
```

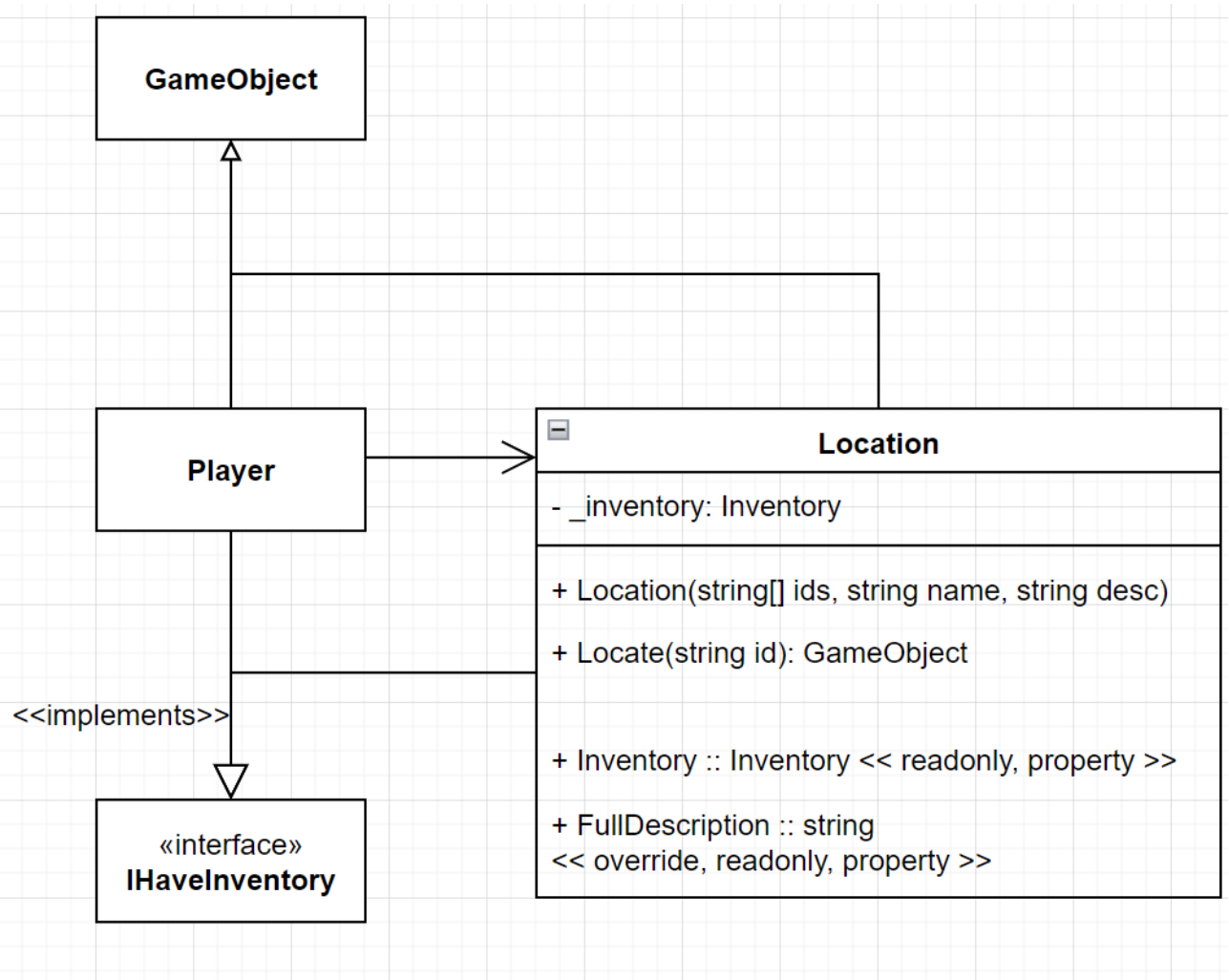
```

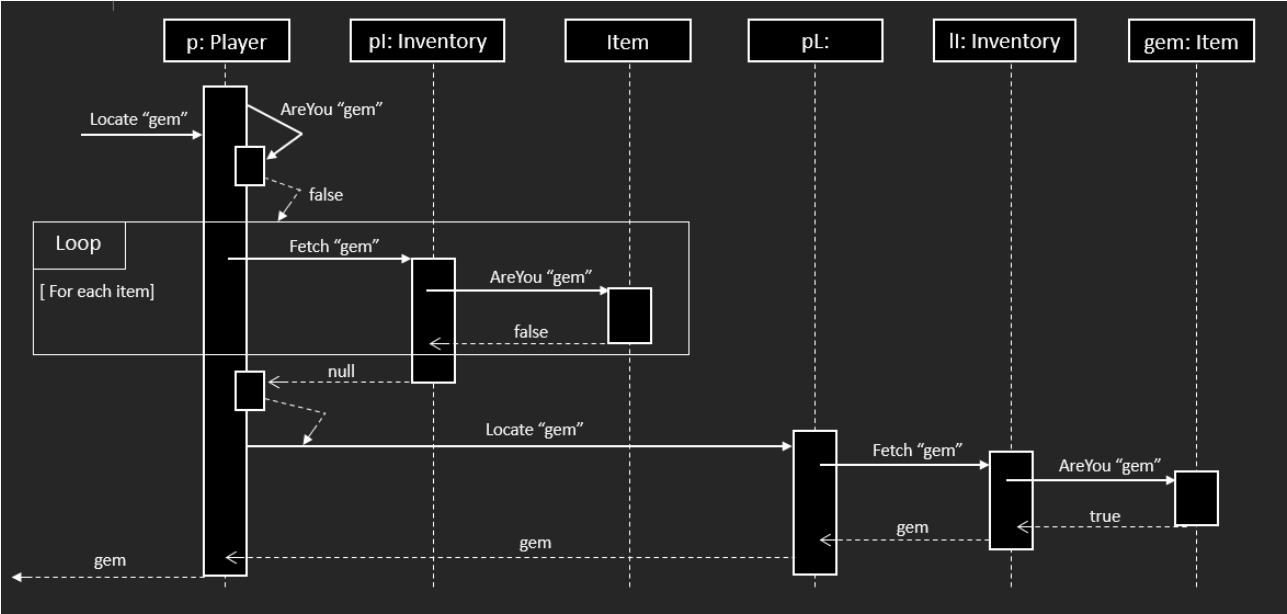
1  using System.Xml.Linq;
2
3  namespace SwinAdventure
4  {
5      public class LookCommandTest
6      {
7          private LookCommand _testLookCommand;
8          private Player _testPlayer;
9          private Bag _testBag;
10
11         private Item _gem;
12         private Item _sword;
13
14         [SetUp]
15         public void SetUp()
16         {
17             _testLookCommand = new LookCommand();
18
19             _gem = new Item(new string[] { "gem" }, "a gem", "This is a gem");
20             _sword = new Item(new string[] { "sword", "bronze" }, "a bronze sword",
↵ "This is a bronze sword");
21
22             _testPlayer = new Player("Trung Kien Nguyen", "I am the player");
23
24             _testBag = new Bag(new string[] { "bag" }, "small bag", "This is a small
↵ bag");
25         }
26
27         [Test]
28         public void TestLookAtMe()
29         {
30             string[] testCommand = new string[] { "look", "at", "inventory" };
31             Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand),
↵ $"You are {_testPlayer.Name}, (I am the player), you are
↵ carrying:\n{_testPlayer.Inventory.ItemList}");
32         }
33
34         [Test]
35         public void TestLookAtGem()
36         {
37             _testPlayer.Inventory.Put(_gem);
38
39             string[] testCommand = new string[] { "look", "at", "gem" };
40             Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand),
↵ "This is a gem");
41             Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand),
↵ _gem.FullDescription);
42         }
43
44         [Test]
45         public void TestLookAtUnk()
46         {
47             string[] testCommand = new string[] { "look", "at", "gem" };

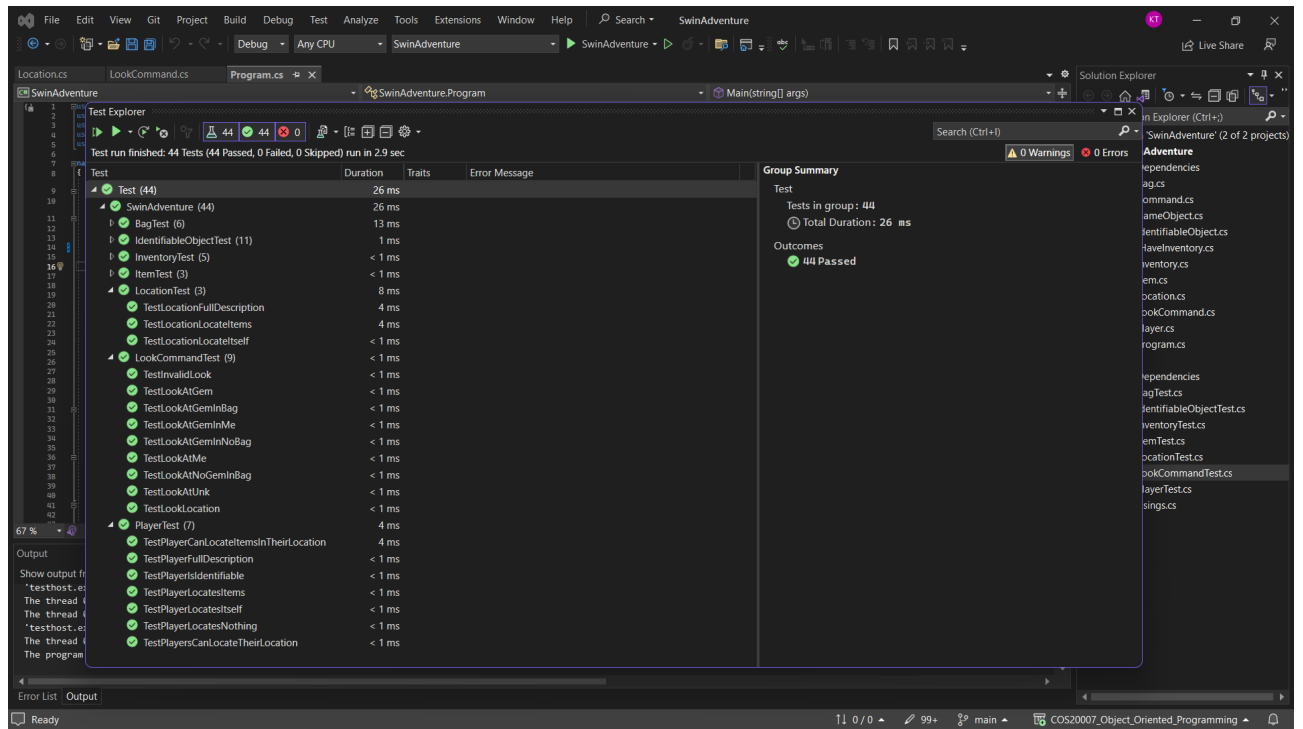
```

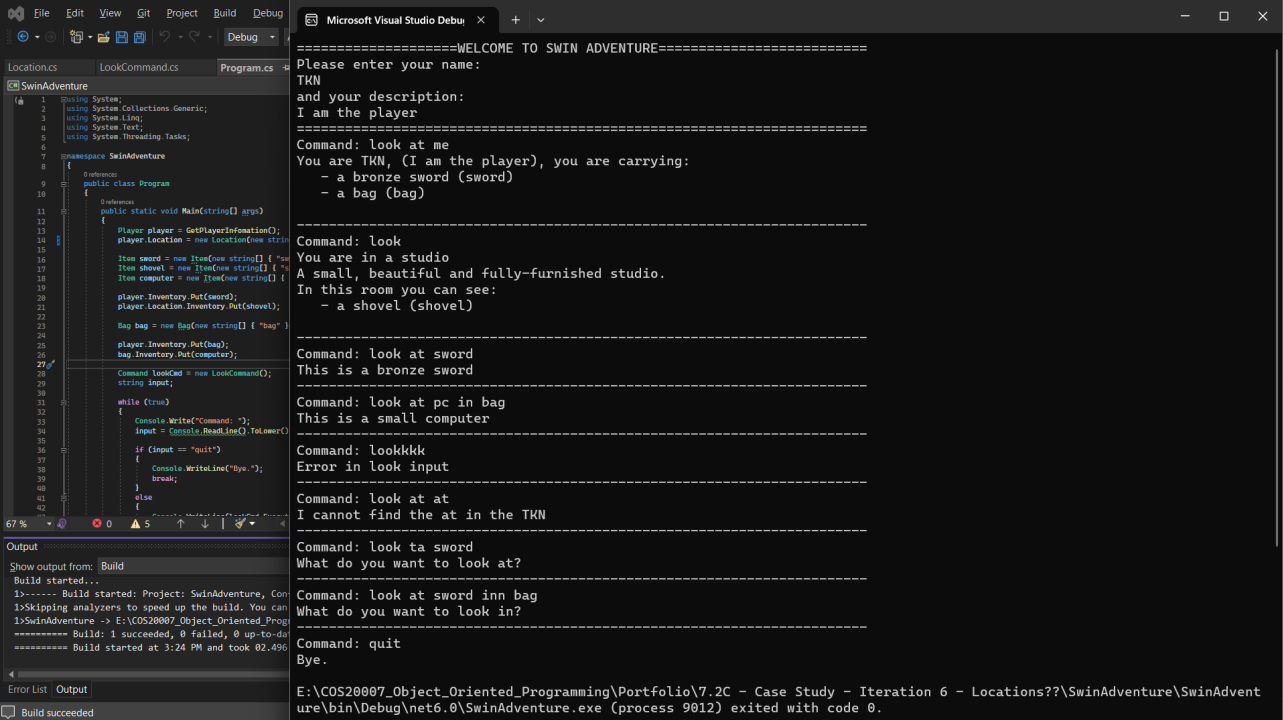
```
48         Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand), $"I
↪ cannot find the {testCommand[2]} in the {_testPlayer.Name}");
49     }
50
51     [Test]
52     public void TestLookAtGemInMe()
53     {
54         _testPlayer.Inventory.Put(_gem);
55
56         string[] testCommand = new string[] { "look", "at", "gem", "in",
↪ "inventory" };
57         Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand),
↪ "This is a gem");
58         Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand),
↪ _gem.FullDescription);
59     }
60
61     [Test]
62     public void TestLookAtGemInBag()
63     {
64         _testBag.Inventory.Put(_gem);
65         _testPlayer.Inventory.Put(_testBag);
66
67         string[] testCommand = new string[] { "look", "at", "gem", "in", "bag" };
68         Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand),
↪ "This is a gem");
69         Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand),
↪ _gem.FullDescription);
70     }
71
72     [Test]
73     public void TestLookAtGemInNoBag()
74     {
75         _testBag.Inventory.Put(_gem);
76         string[] testCommand = new string[] { "look", "at", "gem", "in", "bag" };
77         Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand), $"I
↪ cannot find the {testCommand[4]}");
78     }
79
80     [Test]
81     public void TestLookAtNoGemInBag()
82     {
83         _testPlayer.Inventory.Put(_testBag);
84         string[] testCommand = new string[] { "look", "at", "gem", "in", "bag" };
85         Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand), $"I
↪ cannot find the {testCommand[2]} in the {_testBag.Name}");
86     }
87
88     [Test]
89     public void TestInvalidLook()
90     {
91         string[] testCommand1 = new string[] { "hello", "hi", "howareyou" };
92         string[] testCommand2 = new string[] { "no", "look", "at" };
```

```
93
94     Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand1),
↪ "Error in look input");
95     Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand2),
↪ "Error in look input");
96 }
97
98 [Test]
99 public void TestLookLocation()
100 {
101     _testPlayer.Location = new Location(new string[] { "studio" }, "a
↪ studio", "A small, beautiful and fully-furnished studio.");
102     _testPlayer.Location.Inventory.Put(_sword);
103
104     Assert.AreEqual(_testLookCommand.Execute(_testPlayer, new string[] {
↪ "look" }),
105 ↪ $"You are in {_testPlayer.Location.Name}\nA small, beautiful and
↪ fully-furnished studio.\nIn this room you can
↪ see:\n{_testPlayer.Location.Inventory.ItemList}");
106 }
107 }
108 }
```









The screenshot displays the Microsoft Visual Studio IDE with the 'SwinAdventure' project open. The 'Program.cs' file is visible in the Solution Explorer, showing the main logic of the game. The console window on the right shows the game's execution, including player input and game responses.

```
=====  
=====WELCOME TO SWIN ADVENTURE=====  
Please enter your name:  
TKN  
and your description:  
I am the player  
=====  
Command: look at me  
You are TKN, (I am the player), you are carrying:  
- a bronze sword (sword)  
- a bag (bag)  
-----  
Command: look  
You are in a studio  
A small, beautiful and fully-furnished studio.  
In this room you can see:  
- a shovel (shovel)  
-----  
Command: look at sword  
This is a bronze sword  
-----  
Command: look at pc in bag  
This is a small computer  
-----  
Command: lookkkkk  
Error in look input  
-----  
Command: look at at  
I cannot find the at in the TKN  
-----  
Command: look ta sword  
What do you want to look at?  
-----  
Command: look at sword inn bag  
What do you want to look in?  
-----  
Command: quit  
Bye.  
-----  
E:\COS20007_Object_Oriented_Programming\Portfolio\7.2C - Case Study - Iteration 6 - Locations??\SwinAdventure\SwinAdventure\bin\Debug\net6.0\SwinAdventure.exe (process 9012) exited with code 0.
```

The code in 'Program.cs' includes the following logic:

```
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
  
namespace SwinAdventure  
{  
    public class Program  
    {  
        public static void Main(string[] args)  
        {  
            Player player = GetPlayerInformation();  
            player.Location = new Location(new string[] { "studio" });  
            Item sword = new Item(new string[] { "sword" }, "A small, beautiful and fully-furnished studio.", "bronze sword");  
            Item shovel = new Item(new string[] { "shovel" }, "A small, beautiful and fully-furnished studio.", "shovel");  
            Item computer = new Item(new string[] { "computer" }, "A small, beautiful and fully-furnished studio.", "small computer");  
            player.Inventory.Put(sword);  
            player.Inventory.Put(shovel);  
            Bag bag = new Bag(new string[] { "bag" }, "A small, beautiful and fully-furnished studio.", "bag");  
            player.Inventory.Put(bag);  
            bag.Inventory.Put(computer);  
            Command lookCmd = new LookCommand();  
            string input;  
            while (true)  
            {  
                Console.WriteLine("Command: ");  
                input = Console.ReadLine().ToLower();  
                if (input == "quit")  
                {  
                    Console.WriteLine("Bye.");  
                    break;  
                }  
                else  
                {  
                    lookCmd.Execute(player, input);  
                }  
            }  
        }  
    }  
}
```