```
1  using Iteration1;
2  using System.Numerics;
3
4  namespace LookCommandUnitTests
5  {
6      public class Tests
7      {
8          private Player _player;
9          private Item sword;
10         private Item ak47;
11         private Item gems;
12         private Bag _bag;
13         private LookCommand look;
14
15         [SetUp]
16         public void Setup()
17         {
18             _bag = new Bag(new string[] { "bag", "1" }, "Bag 1", "This is
                 the 1st bag of the player!");
19             _player = new Player("Chien", "A boy with high curiosity");
20             sword = new Item(new string[] { "sword", "melee" }, "bronze
                 sword", "Melee weapon. High damage.");
21             ak47 = new Item(new string[] { "ak47", "gun" }, "ak47", "Gun.
                 High Damage.");
22             gems = new Item(new string[] { "gem" }, " collectible gems",
                 "Using for buying weapons");
23             _player.Inventory.Put(sword);
24             _player.Inventory.Put(ak47);
25             look = new LookCommand();
26         }
27
28         [Test]
29         public void TestLookAtMe()
30         {
31             Assert.That((look.Execute(_player, new string[] { "look at
                 inventory" })), Is.EqualTo(_player.FullDescription));
32             Assert.Pass();
33         }
34
35         [Test]
36         public void TestLookAtGem()
37         {
38             _player.Inventory.Put(gems);
39             Assert.That((look.Execute(_player, new string[] { "look at
                 gem" })), Is.EqualTo(gems.FullDescription));
40             Assert.Pass();
41         }
42
43         [Test]
```

```
44          public void TestLookAtUnk()
45          {
46              Assert.That((look.Execute(_player, new string[] { "look at
                  gem" })), Is.EqualTo("I cannot find the gem"));
47              Assert.Pass();
48          }
49
50          [Test]
51          public void TestLookAtGemInMe()
52          {
53              _player.Inventory.Put(gems);
54              Assert.That((look.Execute(_player, new string[] { "look at gem
                  in inventory" })), Is.EqualTo(gems.FullDescription));
55              Assert.Pass();
56          }
57
58          [Test]
59          public void TestLookAtGemInBag()
60          {
61              _bag.Inventory.Put(gems);
62              _player.Inventory.Put(_bag);
63              Assert.That((look.Execute(_player, new string[] { "look at gem
                  in bag" })), Is.EqualTo(gems.FullDescription));
64              Assert.Pass();
65          }
66
67
68          [Test]
69          public void TestLookAtGemInNoBag()
70          {
71              _bag.Inventory.Put(gems);
72              Assert.That((look.Execute(_player, new string[] { "look at gem
                  in bag" })), Is.EqualTo("I cannot find the bag"));
73              Assert.Pass();
74          }
75
76          [Test]
77          public void TestLookAtNoGemInBag()
78          {
79              _player.Inventory.Put(_bag);
80              Assert.That((look.Execute(_player, new string[] { "look at gem
                  in bag" })), Is.EqualTo("I cannot find the gem in the bag"));
81              Assert.Pass();
82          }
83
84          [Test]
85          public void TestInvalidLook()
86          {
87              Assert.That((look.Execute(_player, new string[] { "look around
```

```
                me" })), Is.EqualTo("What do you want to look at?"));
88              Assert.That((look.Execute(_player, new string[] { "hello me
                friend" })), Is.EqualTo("Error in look input"));
89              Assert.That((look.Execute(_player, new string[] { "look at gem
                a b" })), Is.EqualTo("What do you want to look in?"));
90              Assert.That((look.Execute(_player, new string[] {"look"})),
                Is.EqualTo("I don't know how to look like that!"));
91              Assert.Pass();
92          }
93      }
94  }
```