

```
1 namespace IdentifiableObject
2 {
3     public class Tests
4     {
5         private Player _player;
6         private Player _player_no_bag;
7         private Item _gem;
8         private Bag _bag;
9         private Command _look;
10
11         [SetUp]
12         public void Setup()
13         {
14             _look = new LookCommand();
15             _player = new Player("Duc Thang", "Student");
16             _player_no_bag = new Player("player", "participant");
17             _gem = new Item(new string[] { "gem" }, "a gem", "This is a gem");
18             _bag = new Bag(new string[] { "bag" }, "Thang's bag", "student");
19             _player.Inventory.Put(_bag);
20         }
21
22         [Test]
23         public void TestLookAtMe()
24         {
25             string look_execution = _look.Execute(_player, new string[]
26                 { "look", "at", "inventory" });
27             string output = _player.FullDescription;
28             Assert.AreEqual(look_execution, output);
29         }
30
31         [Test]
32         public void TestLookAtGem()
33         {
34             _player.Inventory.Put(_gem);
35             string look_execution = _look.Execute(_player, new string[]
36                 { "look", "at", "gem" });
37             string output = _gem.FullDescription;
38             Assert.AreEqual(look_execution, output);
39         }
40
41         [Test]
42         public void TestLookAtUnk()
43         {
44             string look_execution = _look.Execute(_player, new string[]
45                 { "look", "at", "gem" });
46             string output = "I can't find the gem";
47             Assert.AreEqual(look_execution, output);
```

```
45     }
46
47     [Test]
48     public void TestLookAtGemInMe()
49     {
50         _player.Inventory.Put(_gem);
51         string look_execution = _look.Execute(_player, new string[] { "look", "at", "gem", "in", "me" });
52         string output = _gem.FullDescription;
53         Assert.AreEqual(look_execution, output);
54     }
55
56     [Test]
57     public void TestLookAtGemInBag()
58     {
59         _bag.Inventory.Put(_gem);
60         string look_execution = _look.Execute(_player, new string[] { "look", "at", "gem", "in", "bag" });
61         string output = _gem.FullDescription;
62         Assert.AreEqual(look_execution, output);
63     }
64
65     [Test]
66     public void TestLookAtGemInNoBag()
67     {
68         string look_execution = _look.Execute(_player_no_bag, new string[] { "look", "at", "bag" });
69         string output = "I can't find the bag";
70         Assert.AreEqual(look_execution, output);
71     }
72
73     [Test]
74     public void TestLookAtNoGemInBag()
75     {
76         string look_execution = _look.Execute(_player, new string[] { "look", "at", "gem", "in", "bag" });
77         string output = "I can't find the gem";
78         Assert.AreEqual(look_execution, output);
79     }
80
81     [Test]
82     public void TestInvalidLook()
83     {
84         Assert.AreEqual(_look.Execute(_player, new string[] { "look", "around" }), "I don't know how to look like that");
85         Assert.AreEqual(_look.Execute(_player, new string[] { "hello", "104776473" }), "I don't know how to look like that");
86         Assert.AreEqual(_look.Execute(_player, new string[] { "look", "at", "Nguyen Duc Thang" }), "I can't find the Nguyen Duc
```

```
    Thang");  
87         }  
88     }  
89 }
```