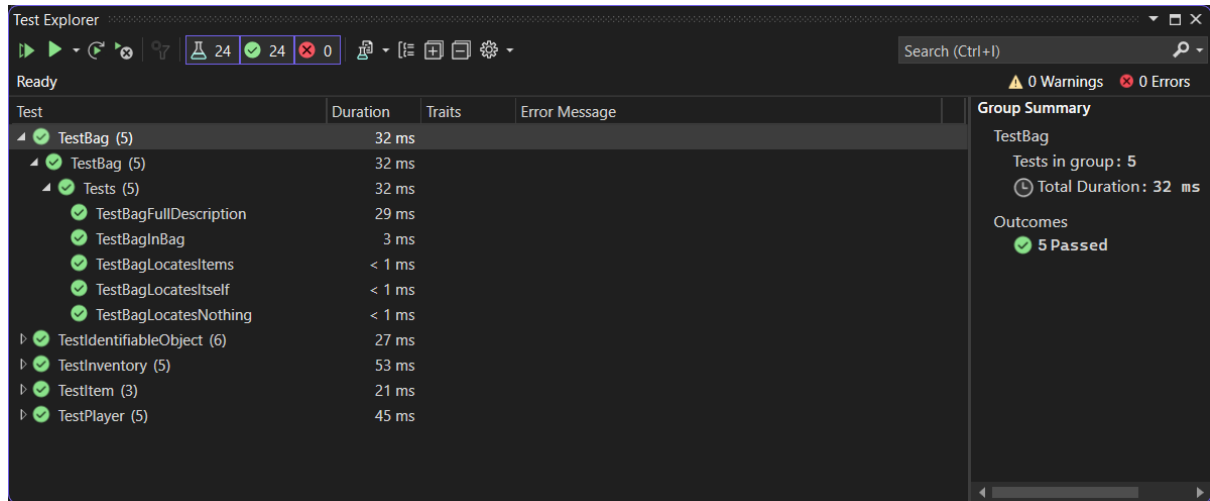


## 5.2P - Case Study - Iteration 3 – Bags

Jayden Kong, 104547242

Tests passing:



```
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 Bag : Item
10    {
11        private Inventory _inventory;
12
13        public string FullDescription
14        {
15            get
16            {
17                return string.Format("In the {0} you can see: {1}", Name,
18                                     Inventory.ItemList);
19            }
20        }
21
22        public Inventory Inventory
23        {
24            get
25            {
26                return _inventory;
27            }
28        }
29
30        public Bag(string[] ids, string name, string desc): base(ids, name,
31                                                                desc)
32        {
33            _inventory = new Inventory();
34        }
35
36        public GameObject Locate(string id)
37        {
38            if (AreYou(id))
39            {
40                return this;
41            }
42            else
43            {
44                return Inventory.Fetch(id);
45            }
46        }
47    }
```

```
1 using SwinAdventure;
2
3 namespace TestBag
4 {
5     public class Tests
6     {
7         [Test]
8         public void TestBagLocatesItems()
9         {
10             Bag testBag = new Bag(new string[] { "bag", "testingBag"},
11                                     "test bag", "this is the test bag's description");
12             Item shovel = new Item(new string[] { "shovel", "spade" }, "a
13                                     shovel", "shovel description");
14             testBag.Inventory.Put(shovel);
15
16             GameObject testLocateShovel = testBag.Locate("shovel");
17             GameObject testShovelRemainsInBag = testBag.Locate("shovel");
18             Assert.That(testLocateShovel, Is.EqualTo(shovel));
19             Assert.That(testShovelRemainsInBag, Is.EqualTo(shovel));
20         }
21
22         [Test]
23         public void TestBagLocatesItself()
24         {
25             Bag testBag = new Bag(new string[] { "bag", "testingBag" },
26                                     "test bag", "this is the test bag's description");
27
28             GameObject testLocateBagID1 = testBag.Locate("bag");
29             GameObject testLocateBagID2 = testBag.Locate("testingBag");
30             Assert.That(testLocateBagID1, Is.EqualTo(testBag));
31             Assert.That(testLocateBagID2, Is.EqualTo(testBag));
32         }
33
34         [Test]
35         public void TestBagLocatesNothing()
36         {
37             Bag testBag = new Bag(new string[] { "bag", "testingBag" },
38                                     "test bag", "this is the test bag's description");
39
40             GameObject testLocateShovel = testBag.Locate("shovel");
41             Assert.That(testLocateShovel, Is.EqualTo(null));
42         }
43
44         [Test]
45         public void TestBagFullDescription()
46         {
47             Bag testBag = new Bag(new string[] { "bag", "testingBag" },
48                                     "test bag", "this is the test bag's description");
49             Item shovel = new Item(new string[] { "shovel", "spade" }, "a
```

```
        shovel", "shovel description");
45     Item bronzeSword = new Item(new string[] { "sword", "bronze
        sword" }, "a bronze sword", "bronze sword description");
46     testBag.Inventory.Put(shovel);
47     testBag.Inventory.Put(bronzeSword);
48
49     string testBagFullDescription = testBag.FullDescription;
50     Assert.That(testBagFullDescription, Is.EqualTo("In the test bag
        you can see: \n a shovel (shovel)\n a bronze sword
        (sword)"));
51 }
52
53 [Test]
54 public void TestBagInBag()
55 {
56     Bag b1 = new Bag(new string[] { "bag", "testingBag1" }, "test
        bag 1", "this is test bag 1's description");
57     Bag b2 = new Bag(new string[] { "bag", "testingBag2" }, "test
        bag 2", "this is test bag 2's description");
58     Item shovel = new Item(new string[] { "shovel", "spade" }, "a
        shovel", "shovel description");
59     Item bronzeSword = new Item(new string[] { "sword", "bronze
        sword" }, "a bronze sword", "bronze sword description");
60     b1.Inventory.Put(shovel);
61     b2.Inventory.Put(bronzeSword);
62     b1.Inventory.Put(b2);
63
64     GameObject testB1LocatesB2 = b1.Locate("testingBag2");
65     GameObject testB1LocatesShovel = b1.Locate("shovel");
66     GameObject testB1LocatesBronzeSword = b1.Locate("sword");
67     Assert.That(testB1LocatesB2, Is.EqualTo(b2));
68     Assert.That(testB1LocatesShovel, Is.EqualTo(shovel));
69     Assert.That(testB1LocatesBronzeSword, Is.EqualTo(null));
70 }
71 }
72 }
```

```
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 IdentifiableObject
10     {
11         private List<string> _identifiers;
12
13         public string FirstID
14         {
15             get
16             {
17                 if (_identifiers.Count > 0)
18                 {
19                     return _identifiers[0];
20                 }
21                 else
22                 {
23                     return "";
24                 }
25             }
26         }
27
28         public IdentifiableObject(string[] idents)
29         {
30             _identifiers = new List<string>();
31             for (int i = 0; i < idents.Length; i++)
32             {
33                 _identifiers.Add(idents[i].ToLower());
34             }
35         }
36
37         public bool AreYou(string id)
38         {
39             bool result = false;
40
41             foreach (string ident in _identifiers)
42             {
43                 if (ident == id.ToLower())
44                 {
45                     result = true;
46                     break;
47                 }
48             }
49         }
50     }
51 }
```

```
50         return result;
51     }
52
53     public void AddIdentifier(string id)
54     {
55         _identifiers.Add(id.ToLower());
56     }
57 }
58 }
59
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel.Design;
4 using System.Linq;
5 using System.Text;
6 using System.Threading.Tasks;
7
8 namespace SwinAdventure
9 {
10     public abstract class GameObject : IdentifiableObject
11     {
12         private string _description;
13         private string _name;
14
15         public string Name
16         {
17             get
18             {
19                 return _name;
20             }
21         }
22
23         public string ShortDescription
24         {
25             get
26             {
27                 return string.Format("{0} ({1})", Name, base.FirstID);
28             }
29         }
30
31         public virtual string FullDescription
32         {
33             get
34             {
35                 return _description;
36             }
37         }
38
39         public GameObject(string[] ids, string name, string desc) : base(ids)
40         {
41             _name = name;
42             _description = desc;
43         }
44     }
45 }
46
```

```
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
10    {
11        private Inventory _inventory;
12
13        public override string FullDescription
14        {
15            get
16            {
17                return string.Format("You are {0}, {1}.\nYou are carrying: {2}", Name, base.FullDescription, Inventory.ItemList);
18            }
19        }
20
21        public Inventory Inventory
22        {
23            get
24            {
25                return _inventory;
26            }
27        }
28
29        public Player(string name, string desc) : base(new string[] { "me", "inventory" }, name, desc)
30        {
31            _inventory = new Inventory();
32        }
33
34        public GameObject Locate(string id)
35        {
36            if (AreYou(id))
37            {
38                return this;
39            }
40            else
41            {
42                return Inventory.Fetch(id);
43            }
44        }
45    }
46 }
47
```



```
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
12            (idents, name, desc) { }
13    }
14 }
```

```
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        private List<Item> _items;
12
13        public string ItemList
14        {
15            get
16            {
17                string itemList = "";
18                foreach (Item item in _items)
19                {
20                    itemList += (string.Format("\n {0}",
21                                                item.ShortDescription));
22                }
23                return itemList;
24            }
25        }
26
27        public Inventory()
28        {
29            _items = new List<Item>();
30        }
31
32        public bool HasItem(string id)
33        {
34            return Fetch(id) != null;
35        }
36
37        public void Put(Item itm)
38        {
39            _items.Add(itm);
40        }
41
42        public Item Take(string id)
43        {
44            foreach (Item item in _items)
45            {
46                if (item.AreYou(id))
47                {
48                    _items.Remove(item);
```

```
49         return item;
50     }
51 }
52 return null;
53 }
54
55 public Item Fetch(string id)
56 {
57     foreach (Item item in _items)
58     {
59         if (item.AreYou(id))
60         {
61             return item;
62         }
63     }
64     return null;
65 }
66
67 }
68 }
69
```

```
1 using SwinAdventure;
2
3 namespace TestIdentifiableObject
4 {
5     public class Tests
6     {
7
8         [Test]
9         public void TestAreYou()
10        {
11            IdentifiableObject myIdents = new IdentifiableObject(new string [] { "fred", "bob" });
12
13            bool fred = myIdents.AreYou("fred");
14            Assert.That(fred, Is.EqualTo(true));
15            bool bob = myIdents.AreYou("bob");
16            Assert.That(bob, Is.EqualTo(true));
17        }
18
19        [Test]
20
21        public void TestNotAreYou()
22        {
23            IdentifiableObject myIdents = new IdentifiableObject(new string [] { "fred", "bob" });
24
25            bool wilma = myIdents.AreYou("wilma");
26            Assert.That(wilma, Is.EqualTo(false));
27            bool boby = myIdents.AreYou("boby");
28            Assert.That(boby, Is.EqualTo(false));
29        }
30
31        [Test]
32
33        public void TestCaseSensitive()
34        {
35            IdentifiableObject myIdents = new IdentifiableObject(new string [] { "fred", "bob" });
36
37            bool fred = myIdents.AreYou("FRED");
38            Assert.That(fred, Is.EqualTo(true));
39            bool bob = myIdents.AreYou("bOB");
40            Assert.That(bob, Is.EqualTo(true));
41        }
42
43        [Test]
44
45        public void TestFirstID()
46        {
```

```
47     IdentifiableObject myIdsents = new IdentifiableObject(new string [] { "fred", "bob" });
48
49     string firstID = myIdsents.FirstID;
50     Assert.That(firstID, Is.EqualTo("fred"));
51 }
52
53 [Test]
54
55 public void TestFirstIDNoIDs()
56 {
57     IdentifiableObject myIdsents = new IdentifiableObject(new string [] {});
58
59     string firstID = myIdsents.FirstID;
60     Assert.That(firstID, Is.EqualTo(""));
61 }
62
63 [Test]
64
65 public void TestAddIDs()
66 {
67     IdentifiableObject myIdsents = new IdentifiableObject(new string [] { "fred", "bob" });
68     myIdsents.AddIdentifier("wilma");
69
70     bool fred = myIdsents.AreYou("fred");
71     Assert.That(fred, Is.EqualTo(true));
72     bool bob = myIdsents.AreYou("bob");
73     Assert.That(bob, Is.EqualTo(true));
74     bool wilma = myIdsents.AreYou("wilma");
75     Assert.That(wilma, Is.EqualTo(true));
76 }
77 }
78 }
```

```
1 using SwinAdventure;
2
3 namespace TestPlayer
4 {
5     public class Tests
6     {
7         [Test]
8         public void TestPlayerIsIdentifiable()
9         {
10             Player p = new Player("Tester", "the mighty test player");
11
12             bool testPMe = p.AreYou("me");
13             bool testPInventory = p.AreYou("inventory");
14             Assert.That(testPMe, Is.EqualTo(true));
15             Assert.That(testPInventory, Is.EqualTo(true));
16         }
17
18         [Test]
19         public void TestPlayerLocatesItems()
20         {
21             Player p = new Player("Tester", "the mighty test player");
22             Item shovel = new Item(new string[] { "shovel", "spade" }, "a shovel", "shovel description");
23             Item bronzeSword = new Item(new string[] { "sword", "bronze sword" }, "a bronze sword", "bronze sword description");
24             p.Inventory.Put(shovel);
25             p.Inventory.Put(bronzeSword);
26
27             GameObject testLocateShovel = p.Locate("shovel");
28             GameObject testLocateBronzeSword = p.Locate("sword");
29             Assert.That(testLocateShovel, Is.EqualTo(shovel));
30             Assert.That(testLocateBronzeSword, Is.EqualTo(bronzeSword));
31         }
32
33         [Test]
34         public void TestPlayerLocatesItself()
35         {
36             Player p = new Player("Tester", "the mighty test player");
37
38             GameObject testLocatePMe = p.Locate("me");
39             GameObject testLocatePInventory = p.Locate("inventory");
40             Assert.That(testLocatePMe, Is.EqualTo(p));
41             Assert.That(testLocatePInventory, Is.EqualTo(p));
42         }
43
44         [Test]
45         public void TestPlayerLocatesNothing()
46         {
47             Player p = new Player("Tester", "the mighty test player");
```

```

...iversity\Year 2\COS20007\5.2P\TestPlayer\UnitTest1.cs 2
48         Item shovel = new Item(new string[] { "shovel", "spade" }, "a  ↗
        shovel", "shovel description");
49         Item bronzeSword = new Item(new string[] { "sword", "bronze  ↗
        sword" }, "a bronze sword", "bronze sword description");
50         p.Inventory.Put(shovel);
51         p.Inventory.Put(bronzeSword);
52
53         GameObject testLocateNothing = p.Locate("nothing");
54         Assert.That(testLocateNothing, Is.EqualTo(null));
55     }
56
57     [Test]
58     public void TestPlayerFullDescription()
59     {
60         Player p = new Player("Tester", "the mighty test player");
61         Item shovel = new Item(new string[] { "shovel", "spade" }, "a  ↗
        shovel", "shovel description");
62         Item bronzeSword = new Item(new string[] { "sword", "bronze  ↗
        sword" }, "a bronze sword", "bronze sword description");
63         p.Inventory.Put(shovel);
64         p.Inventory.Put(bronzeSword);
65
66         string testFullDescription = p.FullDescription;
67         Assert.That(testFullDescription, Is.EqualTo("You are Tester,  ↗
        the mighty test player.\nYou are carrying: \n  a shovel  ↗
        (shovel)\n  a bronze sword (sword)"));
68
69     }
70 }
71 }

```

```
1 using SwinAdventure;
2 using System.Reflection.Metadata;
3
4 namespace TestItem
5 {
6     public class Tests
7     {
8         [Test]
9         public void TestItemIsIdentifiable()
10        {
11            // testing identifiers of Item object
12            Item bronzeSword = new Item(new string[] { "sword", "bronze
13                sword" }, "a bronze sword", "bronze sword description");
14            bool testBronzeSwordID1 = bronzeSword.AreYou("sword");
15            bool testBronzeSwordID2 = bronzeSword.AreYou("bronze sword");
16            Assert.That(testBronzeSwordID1, Is.EqualTo(true));
17            Assert.That(testBronzeSwordID2, Is.EqualTo(true));
18        }
19
20        [Test]
21        public void TestItemShortDescription()
22        {
23            Item bronzeSword = new Item(new string[] { "sword", "bronze
24                sword" }, "a bronze sword", "bronze sword description");
25            string testBronzeSword = bronzeSword.ShortDescription;
26            Assert.That(testBronzeSword, Is.EqualTo("a bronze sword
27                (sword)"));
28        }
29
30        [Test]
31        public void TestItemFullDescription()
32        {
33            Item bronzeSword = new Item(new string[] { "sword", "bronze
34                sword" }, "a bronze sword", "bronze sword description");
35            string testBronzeSword = bronzeSword.FullDescription;
36            Assert.That(testBronzeSword, Is.EqualTo("bronze sword
37                description"));
38        }
39    }
40 }
```



```
1 using SwinAdventure;
2
3 namespace TestInventory
4 {
5     public class Tests
6     {
7
8         [Test]
9         public void TestFindItem()
10        {
11            Item shovel = new Item(new string[] { "shovel", "spade" }, "a
shovel", "shovel description");
12            Item bronzeSword = new Item(new string[] { "sword", "bronze
sword" }, "a bronze sword", "bronze sword description");
13            Inventory testInventory = new Inventory();
14            testInventory.Put(shovel);
15            testInventory.Put(bronzeSword);
16
17            bool testShovel = testInventory.HasItem("shovel");
18            bool testBronzeSword = testInventory.HasItem("sword");
19            Assert.That(testShovel, Is.EqualTo(true));
20            Assert.That(testBronzeSword, Is.EqualTo(true));
21        }
22
23        [Test]
24        public void TestNoItemFind()
25        {
26            Item shovel = new Item(new string[] { "shovel", "spade" }, "a
shovel", "shovel description");
27            Item bronzeSword = new Item(new string[] { "sword", "bronze
sword" }, "a bronze sword", "bronze sword description");
28            Inventory testInventory = new Inventory();
29            testInventory.Put(shovel);
30            testInventory.Put(bronzeSword);
31
32            bool testSmallComputer = testInventory.HasItem("pc");
33            Assert.That(testSmallComputer, Is.EqualTo(false));
34        }
35
36        [Test]
37        public void TestFetchItem()
38        {
39            Item shovel = new Item(new string[] { "shovel", "spade" }, "a
shovel", "shovel description");
40            Item bronzeSword = new Item(new string[] { "sword", "bronze
sword" }, "a bronze sword", "bronze sword description");
41            Inventory testInventory = new Inventory();
42            testInventory.Put(shovel);
43            testInventory.Put(bronzeSword);
```

```
44
45     Item testShovel = testInventory.Fetch("shovel");
46     Item testBronzeSword = testInventory.Fetch("sword");
47     Assert.That(testShovel, Is.EqualTo(shovel));
48     Assert.That(testBronzeSword, Is.EqualTo(bronzeSword));
49 }
50
51 [Test]
52 public void TestTakeItem()
53 {
54     Item shovel = new Item(new string[] { "shovel", "spade" }, "a
55 shovel", "shovel description");
56     Item bronzeSword = new Item(new string[] { "sword", "bronze
57 sword" }, "a bronze sword", "bronze sword description");
58     Inventory testInventory = new Inventory();
59     testInventory.Put(shovel);
60     testInventory.Put(bronzeSword);
61
62     Item testFetchShovel = testInventory.Take("shovel");
63     bool testShovelInInventory = testInventory.HasItem("shovel");
64     Assert.That(testFetchShovel, Is.EqualTo(shovel));
65     Assert.That(testShovelInInventory, Is.EqualTo(false));
66 }
67
68 [Test]
69 public void TestItemList()
70 {
71     Item shovel = new Item(new string[] { "shovel", "spade" }, "a
72 shovel", "shovel description");
73     Item bronzeSword = new Item(new string[] { "sword", "bronze
74 sword" }, "a bronze sword", "bronze sword description");
75     Inventory testInventory = new Inventory();
76     testInventory.Put(shovel);
77     testInventory.Put(bronzeSword);
78
79     string testInventoryList = testInventory.ItemList;
80     Assert.That(testInventoryList, Is.EqualTo("\n a shovel
81 (shovel)\n a bronze sword (sword)"));
82 }
83 }
```