Swinburne University of Technology

COS20007 Object Oriented Programming

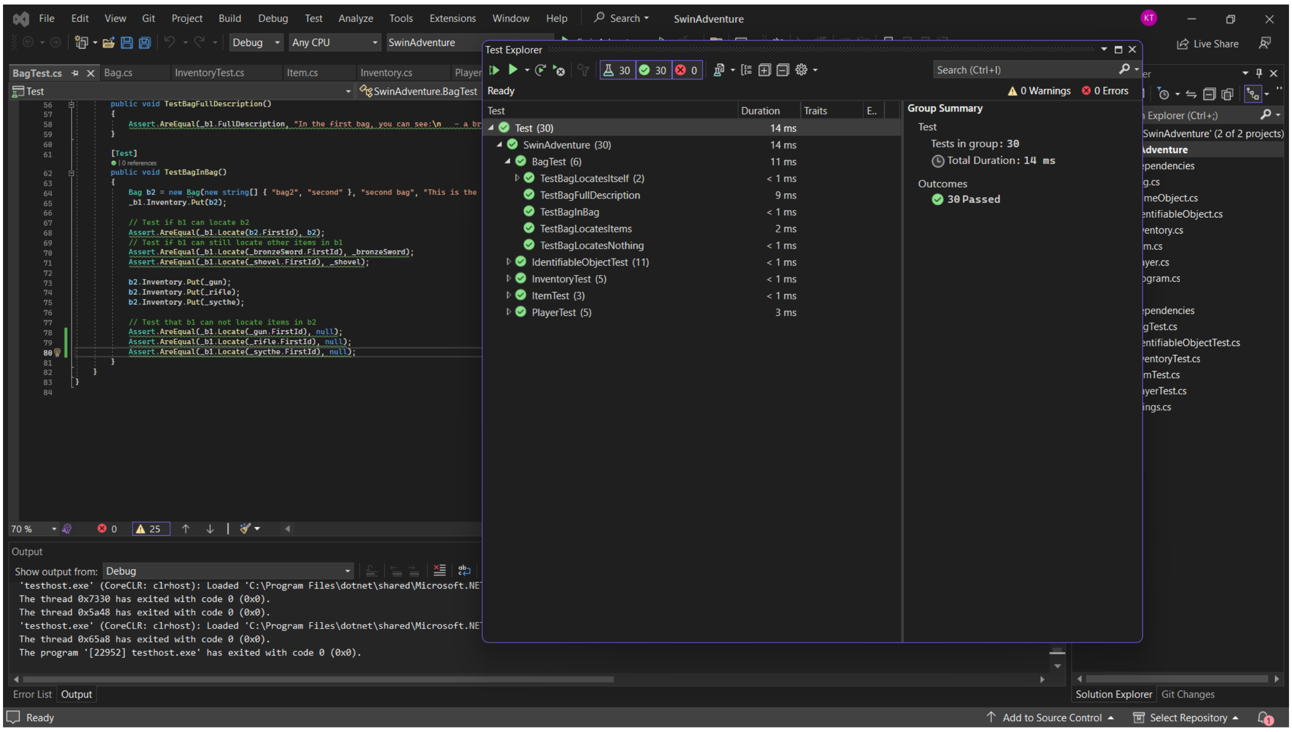
5.2P - Case Study - Iteration 3 - Bags

PDF generated at 22:30 on Tuesday 2nd May, 2023

Produced by OOP Submission Formatter

```
using System;

namespace SwinAdventure
{
    public class Bag : Item
    {
        private Inventory _inventory;

        public Bag(string[] ids, string name, string desc) : base(ids, name, desc)
        {
            _inventory = new Inventory();
        }

        public GameObject Locate(string id)
        {
            if (AreYou(id))
            {
                return this;
            }
            else if (_inventory.HasItem(id))
            {
                return _inventory.Fetch(id);
            }
            return null;
        }

        public override string FullDescription
        {
            get
            {
                return $"In the {this.Name}, you can see:\n" + _inventory.ItemList;
            }
        }

        public Inventory Inventory
        {
            get
            {
                return _inventory;
            }
        }
    }
}
```

```
namespace SwinAdventure
{
    public class BagTest
    {
        private Bag _b1;
        private Item _bronzeSword = new Item(new string[] { "sword", "bronze" }, "a bronze sword", "This is a bronze sword");
        private Item _shovel = new Item(new string[] { "shovel" }, "a shovel", "This is a shovel");
        private Item _computer = new Item(new string[] { "pc", "computer" }, "a small computer", "This is a small computer");
        private Item _gun = new Item(new string[] { "gun", "short" }, "a short gun", "This is a short gun");
        private Item _rifle = new Item(new string[] { "rifle" }, "a rifle", "This is a rifle");
        private Item _sycthe = new Item(new string[] { "sycthe" }, "a sycthe", "This is a sycthe");

        [SetUp]
        public void SetUp()
        {
            _b1 = new Bag(new string[] { "bag1", "first" }, "first bag", "This is the first bag");
            _b1.Inventory.Put(_bronzeSword);
            _b1.Inventory.Put(_shovel);
        }

        [Test]
        public void TestBagLocatesItems()
        {
            Assert.AreEqual(_b1.Locate(_bronzeSword.FirstId), _bronzeSword);
            Assert.AreEqual(_b1.Locate(_shovel.FirstId), _shovel);

            Assert.AreEqual(_b1.Locate(_computer.FirstId), null);

            // Putting "computer" item in the bag's inventory
            _b1.Inventory.Put(_computer);

            // Test if the put item in the bag's inventory
            Assert.AreEqual(_b1.Locate(_computer.FirstId), _computer);
            // Test if others remain in the bag's inventory
            Assert.AreEqual(_b1.Locate(_bronzeSword.FirstId), _bronzeSword);
            Assert.AreEqual(_b1.Locate(_shovel.FirstId), _shovel);
        }

        [TestCase("bag1")]
        [TestCase("first")]
        public void TestBagLocatesItself(string testIdent)
        {
            Assert.AreEqual(_b1.Locate(_b1.FirstId), _b1);
            Assert.AreEqual(_b1.Locate(testIdent), _b1);
        }

        [Test]
        public void TestBagLocatesNothing()
        {
            Assert.AreEqual(_b1.Locate(_gun.FirstId), null);
            Assert.AreEqual(_b1.Locate(_rifle.FirstId), null);
            Assert.AreEqual(_b1.Locate(_sycthe.FirstId), null);
        }

        [Test]
        public void TestBagFullDescription()
        {
            Assert.AreEqual(_b1.FullDescription, "In the first bag, you can see:\n - a bronze sword (sword)\n   - a shovel (shovel)\n");
        }

        [Test]
        public void TestBagInBag()
        {
            Bag b2 = new Bag(new string[] { "bag2", "second" }, "second bag", "This is the second bag");
            _b1.Inventory.Put(b2);

            // Test if b1 can locate b2
            Assert.AreEqual(_b1.Locate(b2.FirstId), b2);
            // Test if b1 can still locate other items in b1
            Assert.AreEqual(_b1.Locate(_bronzeSword.FirstId), _bronzeSword);
            Assert.AreEqual(_b1.Locate(_shovel.FirstId), _shovel);

            b2.Inventory.Put(_gun);
            b2.Inventory.Put(_rifle);
            b2.Inventory.Put(_sycthe);

            // Test that b1 can not locate items in b2
            Assert.AreEqual(_b1.Locate(_gun.FirstId), null);
            Assert.AreEqual(_b1.Locate(_rifle.FirstId), null);
            Assert.AreEqual(_b1.Locate(_sycthe.FirstId), null);
        }
    }
}
```



```
        public void TestBagFullDescription()
        {
            Assert.AreEqual(_b1.FullDescription, "In the first bag, you can see:\n   - a br
        }

        [Test]
        public void TestBagInBag()
        {
            Bag b2 = new Bag(new string[] { "bag2", "second" }, "second bag", "This is the
            _b1.Inventory.Put(b2);

            // Test if b1 can locate b2
            Assert.AreEqual(_b1.Locate(b2.FirstId), b2);
            // Test if b1 can still locate other items in b1
            Assert.AreEqual(_b1.Locate(_bronzeSword.FirstId), _bronzeSword);
            Assert.AreEqual(_b1.Locate(_shovel.FirstId), _shovel);

            b2.Inventory.Put(_gun);
            b2.Inventory.Put(_rifle);
            b2.Inventory.Put(_sycthe);

            // Test that b1 can not locate items in b2
            Assert.AreEqual(_b1.Locate(_gun.FirstId), null);
            Assert.AreEqual(_b1.Locate(_rifle.FirstId), null);
            Assert.AreEqual(_b1.Locate(_sycthe.FirstId), null);
        }
    }
}
```

Test Explorer

30 ✓ 30 ✗ 0

Ready ⚠ 0 Warnings ✗ 0 Errors

```
Test                                        Duration   Traits   E..
✓ Test (30)                                    14 ms
  ✓ SwinAdventure (30)                         14 ms
    ✓ BagTest (6)                              11 ms
      ✓ TestBagLocatesItself (2)             < 1 ms
      ✓ TestBagFullDescription                  9 ms
      ✓ TestBagInBag                         < 1 ms
      ✓ TestBagLocatesItems                     2 ms
      ✓ TestBagLocatesNothing                < 1 ms
    ✓ IdentifiableObjectTest (11)            < 1 ms
    ✓ InventoryTest (5)                      < 1 ms
    ✓ ItemTest (3)                           < 1 ms
    ✓ PlayerTest (5)                            3 ms
```

Group Summary

Test

Tests in group: 30

Total Duration: 14 ms

Outcomes

✓ 30 Passed

Output

Show output from: Debug

```
'testhost.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NE
The thread 0x7330 has exited with code 0 (0x0).
The thread 0x5a48 has exited with code 0 (0x0).
'testhost.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NE
The thread 0x65a8 has exited with code 0 (0x0).
The program '[22952] testhost.exe' has exited with code 0 (0x0).
```