```
using _5._2P;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using NUnit.Framework;
using System.Numerics;



namespace BagTest
{
    [TestFixture]
    public class Tests
    {
        private Item _item1;
        private Item _item2;
        private Bag _bag;

        [SetUp]
        public void Setup()
        {
            _item1 = new Item(new string[] { "sword" }, "Sword", "very stronk sword");
            _item2 = new Item(new string[] { "shield" }, "Shield", "very stronk shield");
            _bag = new Bag(new string[] { "backpack" }, "Backpack", "very gud backpack");
            _bag.Inventory.Put(_item1);
            _bag.Inventory.Put(_item2);
        }

        [TestCase]
        public void TestBagLocatesItems()
        {
            Assert.Pass();
            var locatedItem = _bag.Locate("sword"); //reflect the return type of the method
            Assert.IsNotNull(locatedItem);
            Assert.AreEqual(locatedItem, _item1);
        }

        [TestCase]
        public void TestBagLocatesItself()
        {
            var locatedBag = _bag.Locate("backpack");
            Assert.IsNotNull(locatedBag);
            Assert.AreEqual(locatedBag, _bag);
        }

        [TestCase]
        public void TestBagLocatesNothing()
        {
            var locatedItem = _bag.Locate("money");
            Assert.IsNull(locatedItem);
        }

        [TestCase]
        public void TestBagFullDescription()
        {
            string expectedDescription = "In the Backpack you can see: \tSword (sword)\n\tShield (shield)\n";
            Assert.AreEqual(expectedDescription, _bag.FullDescription);
        }

        [TestCase]
        public void TestBagInBag()
        {
            Bag innerBag = new Bag(new string[] { "innerBag" }, "Inner Bag", "smaller bad");
            Item _item3 = new Item(new string[] { "diamond" }, "Diamond", "very rare diamond");
            innerBag.Inventory.Put(_item3);
            _bag.Inventory.Put(innerBag);

            //Test outer bag can locate inner bag
            var locatedInnerBag = _bag.Locate("innerBag");
            Assert.IsNotNull(locatedInnerBag);
            Assert.AreEqual(innerBag, locatedInnerBag);

            //Test outer bag can locate its item
            var locatedItemInOuterBag = _bag.Locate("sword");
            Assert.IsNotNull(locatedItemInOuterBag);
            Assert.AreEqual(_item1, locatedItemInOuterBag);

            //Test outer bag cannot locate inner bag's item
            var locatedItemInInnerBag = _bag.Locate("diamond");
            Assert.IsNull(locatedItemInInnerBag);
        }
    }
}
```