5.2 iteration 3
Source code
bag.cs

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace SwinAdventure3

{

   public class Bag : Item // inherintance from Item

   {

      Inventory _inventory;


      public Bag(string[] idents, string name, string description) : base(idents, name,
description)

      {

         _inventory = new Inventory();// taking the the list from inventory then initilize it


      }

      public GameObject Locate(string id) //locate

      {

         if (AreYou(id))

         {

            return this;


         }

         else if (_inventory.HasItem(id))
```

```
        {
            return (_inventory.Fetch(id));
        }
        return null;

    }

    public Inventory Inventory //read only property
    {
        get

        { return _inventory; }
    }

  }

}
```

TestBag.cs

```
using SwinAdventure3;
using NUnit.Framework;

namespace IdentifiableObjecttestingBag
{
  public class Tests
  {

    Bag b;
    Bag b1;
```

```
Item Gun = new Item(new string[] { "Gun" }, " a gun", "this is a gun");

Item Katana = new Item(new string[] { "Katana" }, " a Katana", "this is a Katana");

Item Diamond_sword = new Item(new string[] { "Diamond_Sword" }, " a Diamond sword", "this is a Diamond sword");

Item Lazer = new Item(new string[] { "lazer" }, " a lazer", "this is a lazer");


[SetUp]

public void Setup()

{


   b = new Bag(new string[] { "bag" }, "a bag", "this is a bag, that contains a Gun(Gun) and a Katana(katana) in Bag b (Bag b)");

   b1 = new Bag(new string[] { "bag1" }, "a bag1", "this is a bag1, that contains a Diamond_sword(Diamond_sword) and a Lazer(Lazer) in Bag b1 (Bag b1)");

   b.Inventory.Put(Gun); b.Inventory.Put(Katana); //which items goes in bag 1

   b1.Inventory.Put(Diamond_sword); b1.Inventory.Put(Lazer); //which items goes in bag 2

}


[Test]

public void TestBaglocatesItem()

{

   Assert.IsTrue(b.Inventory.HasItem("Gun"));  // see if bag b have these items

   Assert.IsTrue(b.Inventory.HasItem("Katana"));


   Assert.IsTrue(b.Locate(Gun.FirstId) == Gun);  //then locate the items in the bag

   Assert.IsTrue(b.Locate(Katana.FirstId) == Katana);


}
```

```
    [Test]
    public void TestBagLocatesItself()
    {
       Assert.IsTrue(b.Locate(b.FirstId) == b); //bag b
       Assert.IsFalse(b.Locate(b1.FirstId) == b1);  //bag b1
    }
    [Test]
    public void TestBagLocatesNon() //if the bag cant locate its bag
    {
       b.Inventory.take(Gun.FirstId);  //from b and Gun
       Assert.IsFalse(b.Inventory.HasItem("Gun"));
       Assert.IsTrue(b.Inventory.HasItem("Katana"));

       Assert.IsFalse(b.Locate(Gun.FirstId) == Gun);
       Assert.IsTrue(b.Locate(Katana.FirstId) == Katana);
    }
    [Test]
    public void BagFullDesciption()
    {
       string expctout = "this is a bag, that contains a Gun(Gun) and a Katana(katana) in
Bag b (Bag b)";
       Assert.AreEqual(expctout, b.FullDescription);
    }
    [Test]
    public void TestBaginBag()
    {
       b.Inventory.Put(b1);
       Assert.IsTrue(b.Locate(b1.FirstId) == b1);
```

```
        Assert.IsTrue(b.Locate(Gun.FirstId) == Gun);

        Assert.IsFalse(b.Locate(Diamond_sword.FirstId) == Diamond_sword);


        Assert.AreEqual(b.FullDescription, "this is a bag, that contains a Gun(Gun) and a
Katana(katana) in Bag b (Bag b)");

        Assert.AreEqual(b1.FullDescription, "this is a bag1, that contains a
Diamond_sword(Diamond_sword) and a Lazer(Lazer) in Bag b1 (Bag b1)");


    }

  }

}
```

Outcome for unittesting