7.2C- Iteration 6

source code

Player.cs

```
using SwinAdventure4;

using System;

using System.Collections.Generic;

using System.Data.Common;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Xml.Linq;


namespace SwinAdventure4

{

  public class Player : GameObject , IhaveInv //inheritance from bag and player

  {

    private Inventory _inventory;

    private Location _location;

    public Player(string name, string desc) : base(new string[] { "Me", "Inventory " },
name, desc) //overide new info for name and description

    {

      _inventory = new Inventory();


    }




    public GameObject Locate(string id)
```

```
{
    if (AreYou(id)) //first checking locate
    {
        return this;
    }
    GameObject obj = _inventory.Fetch(id);
    if (obj != null) //second checking for object
    {
        return obj;
    }
    if (_location != null) //third checking for location
    {
        obj = _location.Locate(id);
        return obj;
    }
    else
    {
        return null;
    }




}
public override string FullDescription
{
    get
    {
            return $"You are {Name}, " + base.FullDescription + ".\nYou are carrying\n" +
_inventory.ItemList; //display our name is carrying itemlist which it varries between total
list length

        }

    }

    public Inventory Inventory

    {

        get => _inventory;

    }

    public Location Location

    {

        get => _location;

        set => _location = value; //get and set value into _location

    }


  }


}
```

Location.cs

```
using SwinAdventure4;

using System;

using System.Collections.Generic;

using System.Data.Common;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Xml.Linq;


namespace SwinAdventure4
```

```
{
  public class Location : GameObject, IhaveInv //inheritance from bag and player
  {
    private Inventory _inventory;

    public Location(string name, string desc) : base(new string[] { "Location"
},name,desc) //overide new info for name and description
    {
      _inventory = new Inventory();

    }



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

    }
    public override string FullDescription
    {
      get
      {
        StringBuilder description = new StringBuilder();
```

```
            description.AppendLine($"You are in {Name}."); //avoid stack overflow and
duplication

            description.AppendLine("Items at this location:");

            description.AppendLine(_inventory.ItemList); // Ensure ItemList is a simple list,
not calling FullDescription


            return description.ToString();

        }

      }

      public Inventory Inventory

      {

        get => _inventory;

      }



  }


}
```

LookCommand.cs

```
using System;

using System.Collections;

using System.Collections.Generic;

using System.ComponentModel;

using System.Linq;

using System.Runtime.CompilerServices;

using System.Text;

using System.Threading.Tasks;

using System.Xml.Linq;
```

```
namespace SwinAdventure4

{

  public class LookCommand : Command

  {

    public LookCommand() : base(new string[] { "Look" })

    {


    }


    public override string Execute(Player p, string[] text)

    {

      IhaveInv _container;

      string _itemid;

      string error = "Error in look input.";


      if (text[0].ToLower() != "look")

        return error;


      switch (text.Length)

      {

        case 1:

          _container = p;

          _itemid = "location";

          break;


        case 3:

          if (text[1].ToLower() != "at")

            return "What do you want to look at?";
          _container = p;
          _itemid = text[2];
          break;

        case 5:
          _container = FetchContainer(p, text[4]);
          if (_container == null)
            return "Could not find " + text[4];
          _itemid = text[2];
          break;

        default:
          return error;
      }
      return LookAtLn(_itemid, _container);
    }



    private IhaveInv FetchContainer(Player p, string ContainerId) //use ihaveInv fetch
from thr
    {
      return p.Locate(ContainerId) as IhaveInv;
    }

    private string LookAtLn(string thingId, IhaveInv container)
    {
      GameObject item = container.Locate(thingId) as GameObject;
      if (item != null)
      {
        if (item is Bag bag)
        {
          StringBuilder contentsDescription = new StringBuilder(item.FullDescription);
          contentsDescription.Append("\nIt contains:\n");
          contentsDescription.Append(bag.Inventory.ItemList); // Use ItemList property
here

          return contentsDescription.ToString();
        }
        return item.FullDescription;
      }
      return "Couldn't find";
    }


}
}
```

program.cs

```
using SwinAdventure4;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Xml.Linq;

namespace SwinAdventure4
{
```

```
public class Program
{
    static void LookCommandExe(Command l, string Input, Player player)
    {
        Console.WriteLine(l.Execute(player, Input.Split()));
    }

    static void Main(string[] args)
    {
        //Greeting + info
        string name, desc;
        string help = "-look\n\nGetting list of item:\n-look at me\n-look at bag\n\nGetting
item description:\nlook at {item}\nlook at {item} in me\nlook at {item} in bag\n\n";
        Console.WriteLine(help);




        //Setting up player


        Console.Write("Setting up player:\nPlayer Name: ");
        name = Console.ReadLine();
        Console.Write("Player Description: ");
        desc = Console.ReadLine();
        Player player = new Player(name, desc);

        //setting a location
        Location Myroom = new Location("MyRoom", $"This is my Room");
        player.Location = Myroom;
        //Setting up list of items


        Item bed = new Item(new string[] { "Bed" }, "a Bed", "This is a Bed");

        Item PC = new Item(new string[] { "PC" }, "a PC", "This is a PC");

        Item Nintendo = new Item(new string[] { "Nintendo" }, "a Nintendo", "This is a
Nintendo");

        Myroom.Inventory.Put(bed);

        Myroom.Inventory.Put(PC);

        Myroom.Inventory.Put(Nintendo);



        Item shovel = new Item(new string[] { "shovel" }, "a shovel", "This is a shovel"); //
declare two items

        Item sword = new Item(new string[] { "sword" }, "a sword", "This is a sword");

        player.Inventory.Put(shovel); //put 2 item in iventory

        player.Inventory.Put(sword);




        Bag bag = new Bag(new string[] { $"bag" }, $"{player.Name}'s bag", $"This is
{player.Name}'s bag"); //create a bag

        player.Inventory.Put(bag); //place item in bag

        Item diamond = new Item(new string[] { "diamond" }, "a diamond", "This is a
diamond");

        bag.Inventory.Put(diamond);


        string _input;

        Command l = new LookCommand();
        while (true)

        {

            Console.Write("Command: ");

            _input = Console.ReadLine();

            if (_input == "quit")

            {

                break;

            }

            else if (_input == "help")

            {

                Console.Write(help);

            }

            else

            {

                LookCommandExe(l, _input, player);

            }


        }



    }



}
```

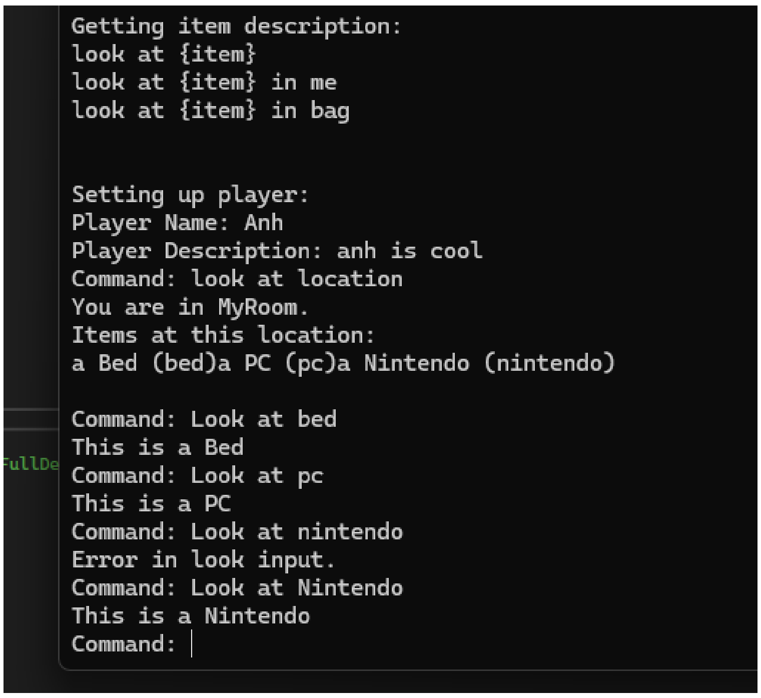
```
}
Testunit
using System;

using System.Collections.Generic;

using System.Linq;

using System.Reflection.Emit;

using System.Text;

using System.Threading.Tasks;

using NUnit.Framework;

using SwinAdventure4;



namespace IdentifiableObjectTestingLocation

{


  public class TestLocation

  {

    Player p = new Player("Anh", "This is Anh");

    Location l = new Location("MyRoom", "This is my room");

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


    [SetUp]

    public void Setup()

    {


    }


    [Test]

    public void TestLookCommand()
```

```
{
   p.Location = l;
   bool actual = l.AreYou("Location");
   Assert.IsTrue(actual);
}
[Test]
public void TestNotLookCommand()
{
   p.Location = l;
   bool actual = l.AreYou("hi");
   Assert.IsFalse(actual);
}

[Test]
public void TestPlayerHasLocation()
{
   p.Location = l;
   GameObject expect = l;
   GameObject actual = p.Locate("location");
   Assert.AreEqual(expect, actual);
}

[Test]
public void TestLocationLocateTest()
{
   l.Inventory.Put(sword);
   GameObject expect = sword;
   GameObject actual = l.Locate("sword");
```

```
            Assert.AreEqual (expect, actual);

        }


    }

}
```

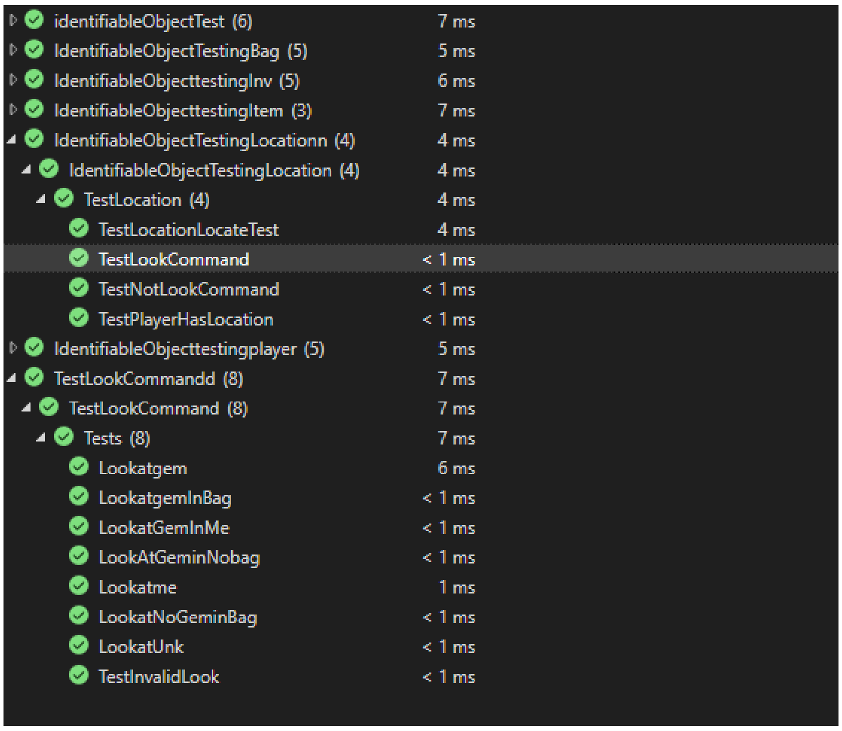
Outputs:

Program.cs



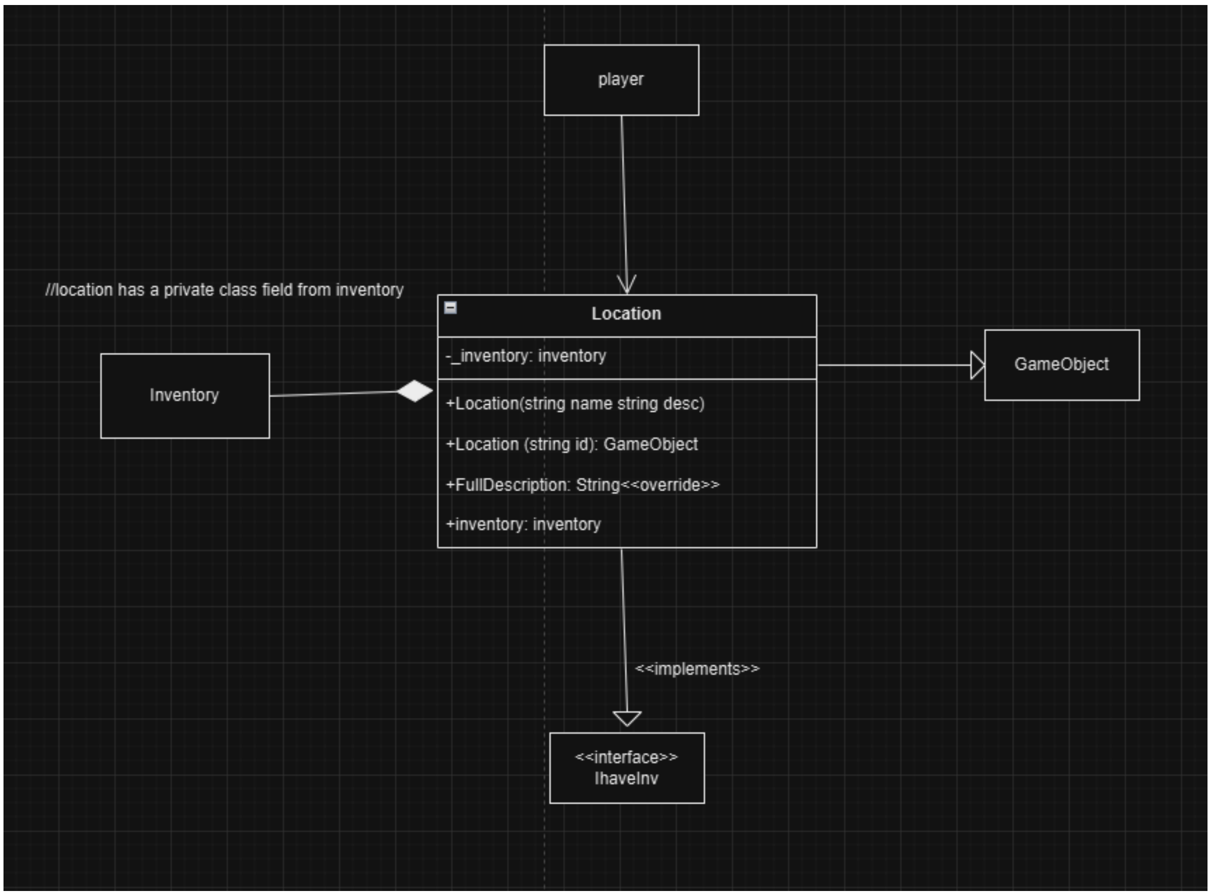
```
Getting item description:
look at {item}
look at {item} in me
look at {item} in bag


Setting up player:
Player Name: Anh
Player Description: anh is cool
Command: look at location
You are in MyRoom.
Items at this location:
a Bed (bed)a PC (pc)a Nintendo (nintendo)

Command: Look at bed
This is a Bed
Command: Look at pc
This is a PC
Command: Look at nintendo
Error in look input.
Command: Look at Nintendo
This is a Nintendo
Command:
```

UnitTesting

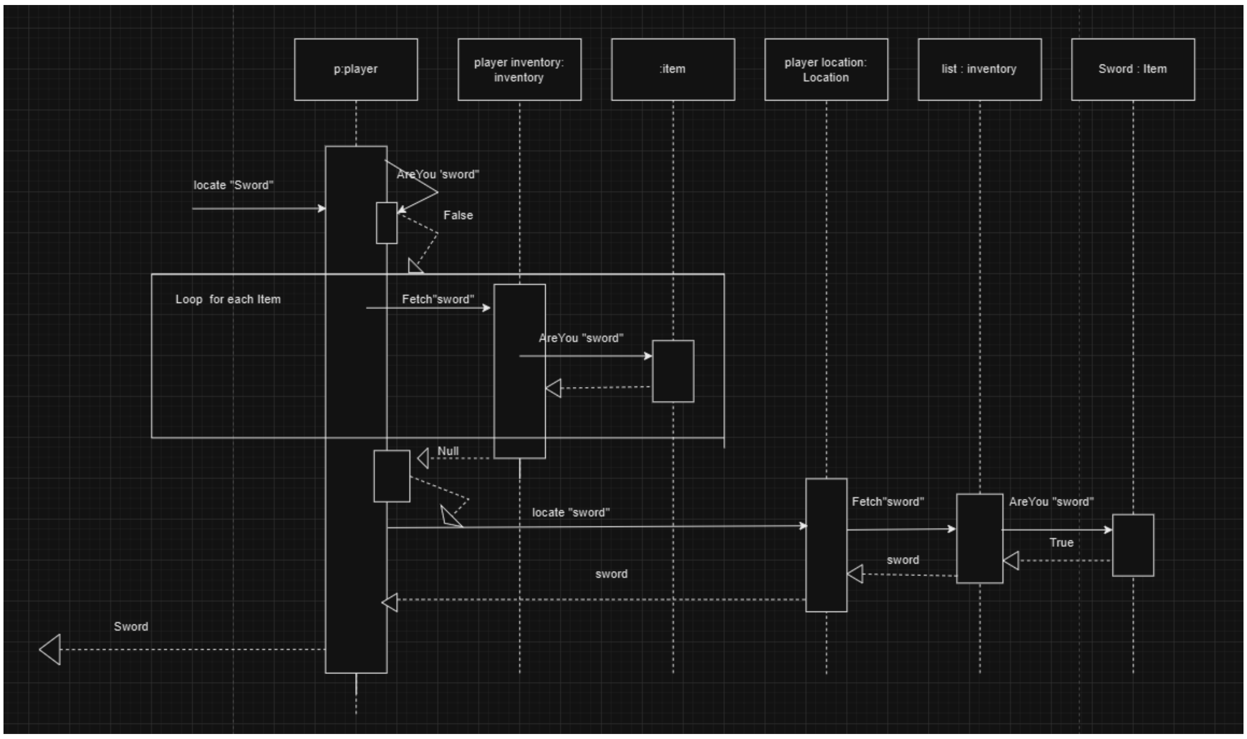


```
◁ ✓ identifiableObjectTest (6)                          7 ms
◁ ✓ IdentifiableObjectTestingBag (5)                    5 ms
◁ ✓ IdentifiableObjecttestingInv (5)                    6 ms
◁ ✓ IdentifiableObjecttestingItem (3)                   7 ms
◢ ✓ IdentifiableObjectTestingLocationn (4)              4 ms
  ◢ ✓ IdentifiableObjectTestingLocation (4)             4 ms
    ◢ ✓ TestLocation (4)                                4 ms
        ✓ TestLocationLocateTest                        4 ms
        ✓ TestLookCommand                             < 1 ms
        ✓ TestNotLookCommand                          < 1 ms
        ✓ TestPlayerHasLocation                       < 1 ms
◁ ✓ IdentifiableObjecttestingplayer (5)                 5 ms
◢ ✓ TestLookCommandd (8)                                7 ms
  ◢ ✓ TestLookCommand (8)                               7 ms
    ◢ ✓ Tests (8)                                       7 ms
        ✓ Lookatgem                                     6 ms
        ✓ LookatgemInBag                              < 1 ms
        ✓ LookatGemInMe                               < 1 ms
        ✓ LookAtGeminNobag                            < 1 ms
        ✓ Lookatme                                      1 ms
        ✓ LookatNoGeminBag                            < 1 ms
        ✓ LookatUnk                                   < 1 ms
        ✓ TestInvalidLook                             < 1 ms
```

UML Diagram


player
//location has a private class field from inventory
Location
-_inventory: inventory
+Location(string name string desc)
+Location (string id): GameObject
+FullDescription: String<<override>>
+inventory: inventory
Inventory
GameObject
<<implements>>
<<interface>>
IhaveInv

Sequence Diagram


p:player
player inventory: inventory
:item
player location: Location
list : inventory
Sword : Item
locate "Sword"
AreYou 'sword"
False
Loop for each Item
Fetch"sword"
AreYou "sword"
Null
locate "sword"
Fetch"sword"
AreYou "sword"
True
sword
sword
Sword