COS 20007
Task 4.2

Duc Thuan Tran
104330455

I. Swin-Adventure

1. IdentifiableObject.cs

```
using System;
namespace Swin_Adventure
{
	public class IdentifiableObject
	{
		private List<string> _identifiers;

		public IdentifiableObject(string[] idents)
		{
			_identifiers = new List<string>(idents);
			_identifiers.AddRange(idents);
		}

		public bool AreYou(string id)
		{
			return _identifiers.Contains(id.ToLower());
		}

		public string FirstId
		{
			get
			{
				if (_identifiers.Count == 0)
				{
					return "";
				}
				return _identifiers[0];
			}
		}

		public void AddIdentifier(string id)
		{
			_identifiers.Add(id.ToLower());
		}
	}
}
```

2. GameObject.cs

```
using System;
using System.Xml.Linq;
```

```
namespace Swin_Adventure
{
        public class GameObject : IdentifiableObject
        {
    private string _description;
    private string _name;

    public GameObject(string[] ids, string name, string description) : base(ids)
    {
      _description = description;
      _name = name;
    }

    public string Name
                {
                        get { return _name.ToLower(); }
                }

                public string ShortDescription
                {
      get { return $"a {_name.ToLower()} ({FirstId.ToLower()})"; }
    }

    public virtual string FullDescription
                {
                        get { return _description; }
                }
        }
}
```

3. Inventory.cs

```
using System;
namespace Swin_Adventure
{
        public class Inventory
        {
                private List<Item> _items;
                public Inventory()
                {
                        _items = new List<Item>();
                }

                public bool HasItem(string id)
                {
                        foreach (Item itm in _items)
                        {
```

```
                        if (itm.AreYou(id))
                        {
                                return true;
                        }
                }
                return false;
        }

        public void Put(Item itm)
        {
                _items.Add(itm);
        }

public Item Take(string id)
{
   Item itm = Fetch(id);

   if (itm != null)
   {
      _items.Remove(itm);
   }

   return itm;
}

public Item Fetch(string id)
        {
                foreach (Item itm in _items)
                {
                        if (itm.AreYou(id))
                        {
                                return itm;
                        }
                }
                return null;
        }

        public string ItemList
        {
                get
                {
                        string list = "";
      foreach (Item item in _items)
      {
         list += "\t" + "a " + item.Name + " (" + item.FirstId + ")\n";
      }
      return list;
```

```
        }
                }
         }
}
```

4. Item.cs

```
using System;

namespace Swin_Adventure
{
   public class Item : GameObject
   {
      public Item(string[] idents, string name, string description) : base(idents, name,
description)
      {

      }
   }
}
```

5. Player.cs

```
using System;
namespace Swin_Adventure
{
          public class Player : GameObject
          {
      private Inventory _inventory;

      public Player(string name, string description) : base(new string[] { "me",
"inventory" }, name, description)
      {
         _inventory = new Inventory();
      }

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

                public override string FullDescription
                {
                        get
                        {
            return "You are " + Name + ", " + base.FullDescription + ".\n"
                                + "You are carrying:\n" + Inventory.ItemList;
```

```
        }
                }

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

6. Program.cs

```
namespace Swin_Adventure;

class Program
{
   static void Main(string[] args)
   {
      Console.WriteLine("Duc Thuan Tran - 104330455");
   }
}
```

II. Swin-Adventure Test

1. IdentifiableObjectTest.cs

```
using NUnit.Framework;
using Swin_Adventure;

namespace IdentifiableObjectTest
{

   internal class Tests
   {
      private IdentifiableObject _test1;
      private IdentifiableObject _test2;
      private IdentifiableObject _test3;
      private IdentifiableObject _test4;
      private IdentifiableObject _test5;
      private IdentifiableObject _test6;

      [SetUp]
      public void Setup()
      {
         _test1 = new IdentifiableObject(new string[] { "fred", "bob" });
         _test2 = new IdentifiableObject(new string[] { "fred", "bob" });
         _test3 = new IdentifiableObject(new string[] { "fred", "bob" });
         _test4 = new IdentifiableObject(new string[] { "fred", "bob" });
```

```
            _test5 = new IdentifiableObject(new string[] { });
            _test6 = new IdentifiableObject(new string[] { "fred", "bob" });
        }

        [Test]
        public void TestAreYou()
        {
            Assert.IsTrue(_test1.AreYou("fred"));
            Assert.IsTrue(_test1.AreYou("bob"));
        }

        [Test]
        public void TestNotAreYou()
        {
            Assert.IsFalse(_test2.AreYou("wilma"));
            Assert.IsFalse(_test2.AreYou("boby"));
        }

        [Test]
        public void TestCaseSensitive()
        {
            Assert.IsTrue(_test3.AreYou("FRED"));
            Assert.IsTrue(_test3.AreYou("bOB"));
        }

        [Test]
        public void TestFirstID()
        {
            Assert.AreEqual("fred", _test4.FirstId);
        }

        [Test]
        public void TestFirstIdWithNoIDs()
        {
            Assert.AreEqual("", _test5.FirstId);
        }

        [Test]
        public void TestAddID()
        {
            _test6.AddIdentifier("wilma");
            Assert.IsTrue(_test6.AreYou("fred"));
            Assert.IsTrue(_test6.AreYou("bob"));
            Assert.IsTrue(_test6.AreYou("wilma"));
        }
    }
}
```

2. Inventory.cs

```
using System;
using Swin_Adventure;

namespace SwinAdventureTest
{
   [TestFixture]
   public class InventoryTest
         {
                  private Inventory _inventoryTest;
                  private Item _weaponTest;
                  private Item _armorTest;

                  [SetUp]
                  public void SetUp()
                  {
                           _inventoryTest = new Inventory();
                           _weaponTest = new Item(new string[] { "weapon" }, "sword",
"this is a Excalibur");
         _armorTest = new Item(new string[] { "armor" }, "shield", "this is a shield");

         _inventoryTest.Put(_weaponTest);
         _inventoryTest.Put(_armorTest);
      }

      [Test]
      public void TestFindItem()
      {
         Assert.IsTrue(_inventoryTest.HasItem("weapon"));
         Assert.IsTrue(_inventoryTest.HasItem("armor"));
      }

      [Test]
      public void TestNoItemFind()
      {
         Assert.IsFalse(_inventoryTest.HasItem("axe"));
         Assert.IsFalse(_inventoryTest.HasItem("helmet"));
      }

      [Test]
      public void TestFetchItem()
      {
         Assert.IsTrue(_weaponTest == _inventoryTest.Fetch("weapon"));
         Assert.IsTrue(_inventoryTest.HasItem("weapon"));

         Assert.IsTrue(_armorTest == _inventoryTest.Fetch("armor"));
         Assert.IsTrue(_inventoryTest.HasItem("armor"));
```

```
        }

        [Test]
        public void TestTakeItem()
        {
            Assert.IsTrue(_weaponTest == _inventoryTest.Take("weapon"));
            Assert.IsFalse(_inventoryTest.HasItem("weapon"));

            Assert.IsTrue(_armorTest == _inventoryTest.Take("armor"));
            Assert.IsFalse(_inventoryTest.HasItem("armor"));
        }

        [Test]
        public void TestItemList()
        {
            Assert.IsTrue(_inventoryTest.ItemList.Replace("\t", "") == "a sword
(weapon)\na shield (armor)\n");
        }
    }
}
```

3. Item.cs

```
using System;
using Swin_Adventure;

namespace SwinAdventureTest
{
    [TestFixture]
    public class ItemTest
          {
                private Item _itemTest;


        [SetUp]
        public void Setup()
        {
            _itemTest = new Item(new string[] { "weapon" }, "sword", "This is an
Excalibur");

        }

        [Test]
        public void TestItemIsIdentifiable()
        {
            Assert.IsTrue(_itemTest.AreYou("weapon"));
        }
```

```
        [Test]
        public void TestShortDescription()
        {
            Assert.IsTrue(_itemTest.ShortDescription == "a sword (weapon)");
        }

        [Test]
        public void TestFullDescription()
        {
            Assert.IsTrue(_itemTest.FullDescription == "This is an Excalibur");
        }
    }
}
```

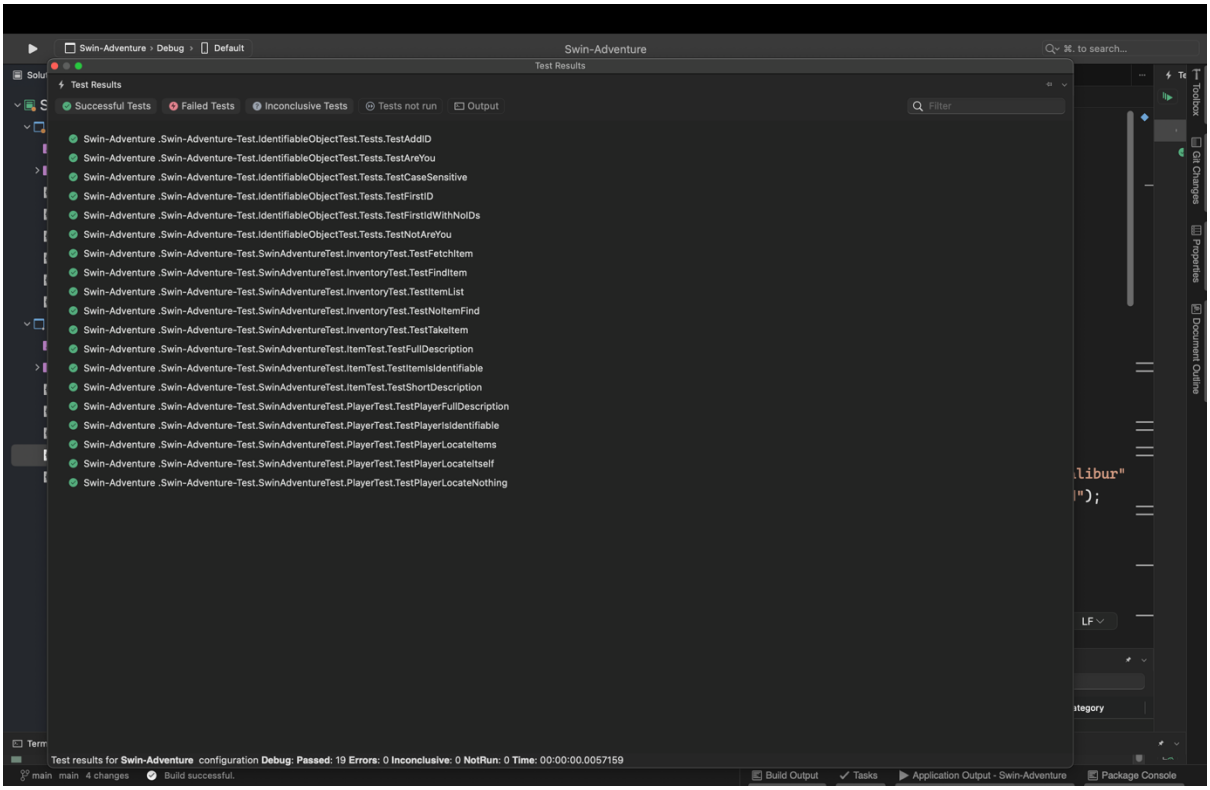
4. Player.cs

```
using System;
using Swin_Adventure;

namespace SwinAdventureTest
{
    [TestFixture]
    public class PlayerTest
    {
        private Player _playerTest;
        private Item _weaponTest;
        private Item _armorTest;

        [SetUp]
        public void Setup()
        {
            _playerTest = new Player("thuan", "dan choi");
            _weaponTest = new Item(new string[] { "weapon" }, "sword", "this is an
Excalibur");
            _armorTest = new Item(new string[] { "armor" }, "shield", "this is a shield");



            _playerTest.Inventory.Put(_weaponTest);
            _playerTest.Inventory.Put(_armorTest);
        }

        [Test]
        public void TestPlayerIsIdentifiable()
        {
            Assert.IsTrue(_playerTest.AreYou("me"));
            Assert.IsTrue(_playerTest.AreYou("inventory"));
        }
```

```
        [Test]
        public void TestPlayerLocateItems()
        {
            Assert.IsTrue(_playerTest.Locate("weapon") == _weaponTest);
            Assert.IsTrue(_playerTest.Locate("armor") == _armorTest);

            Assert.IsTrue(_playerTest.Inventory. HasItem("weapon"));
            Assert.IsTrue(_playerTest.Inventory.HasItem("armor"));
        }

        [Test]
        public void TestPlayerLocateItself()
        {
            Assert.IsTrue(_playerTest == _playerTest.Locate("me"));
            Assert.IsTrue(_playerTest == _playerTest.Locate("inventory"));
        }

        [Test]
        public void TestPlayerLocateNothing()
        {
            Assert.IsTrue(_playerTest.Locate("helmet") == null);
        }

        [Test]
        public void TestPlayerFullDescription()
        {
            Assert.IsTrue(_playerTest.FullDescription == "You are thuan, dan choi.\nYou
    are carrying:\n\ta sword (weapon)\n\ta shield (armor)\n");
        }
    }
}
```

III. Image

1. NUnit test run


Swin-Adventure › Debug › Default
Swin-Adventure
Test Results
Test Results
Successful Tests
Failed Tests
Inconclusive Tests
Tests not run
Output
Filter
Swin-Adventure .Swin-Adventure-Test.IdentifiableObjectTest.Tests.TestAddID
Swin-Adventure .Swin-Adventure-Test.IdentifiableObjectTest.Tests.TestAreYou
Swin-Adventure .Swin-Adventure-Test.IdentifiableObjectTest.Tests.TestCaseSensitive
Swin-Adventure .Swin-Adventure-Test.IdentifiableObjectTest.Tests.TestFirstID
Swin-Adventure .Swin-Adventure-Test.IdentifiableObjectTest.Tests.TestFirstIdWithNoIDs
Swin-Adventure .Swin-Adventure-Test.IdentifiableObjectTest.Tests.TestNotAreYou
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.InventoryTest.TestFetchItem
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.InventoryTest.TestFindItem
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.InventoryTest.TestItemList
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.InventoryTest.TestNoItemFind
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.InventoryTest.TestTakeItem
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.ItemTest.TestFullDescription
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.ItemTest.TestItemIsIdentifiable
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.ItemTest.TestShortDescription
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.PlayerTest.TestPlayerFullDescription
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.PlayerTest.TestPlayerIsIdentifiable
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.PlayerTest.TestPlayerLocateItems
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.PlayerTest.TestPlayerLocateItself
Swin-Adventure .Swin-Adventure-Test.SwinAdventureTest.PlayerTest.TestPlayerLocateNothing
Test results for Swin-Adventure configuration Debug: Passed: 19 Errors: 0 Inconclusive: 0 NotRun: 0 Time: 00:00:00.0057159
Build successful.
Build Output
Tasks
Application Output - Swin-Adventure
Package Console
Toolbox
Git Changes
Properties
Document Outline