**6.1. All cs. Program files are included in Submission.**

**LookCommand Test Code:**

[Test]

public void LookAtMe()

{

Player player = new Player("Tan", "A player");

player.Inventory.Put(item1);

player.Inventory.Put(item2);

LookCommand LookCommand = new LookCommand();

string expectedDescription = "You are Tan A player\nYou are carrying:\nsword (sword)\nshield (shield)\n";

string testDescription = LookCommand.Execute(player, new string[] { "look", "at", "me" });

Assert.That(testDescription, Is.EqualTo(expectedDescription));

}

[Test]

public void LookAtGem()

{

Player player = new Player("Tan", "A player");

player.Inventory.Put(item4);

LookCommand LookCommand = new LookCommand();

string expectedDescription = "a gem";

string testDescription = LookCommand.Execute(player, new string[] { "look", "at", "gem" });

Assert.That(testDescription, Is.EqualTo(expectedDescription));

}

[Test]

public void LookAtUnk()

{

Player player = new Player("Tan", "A player");

LookCommand LookCommand = new LookCommand();

string expectedDescription = "I can't find the gem in the Tan";

string testDescription = LookCommand.Execute(player, new string[] { "look", "at", "gem" });

Assert.That(testDescription, Is.EqualTo(expectedDescription));

}

[Test]

public void LookAtGemInBag()

{

Player player = new Player("Tan", "A player");

Bag backpack = new Bag(new string[] { "backpack" }, "backpack", "a backpack");

player.Inventory.Put(backpack);

backpack.Inventory.Put(item4);

LookCommand LookCommand = new LookCommand();

string expectedDescription = "a gem";

string testDescription = LookCommand.Execute(player, new string[] { "look", "at", "gem", "in", "backpack" });

Assert.That(testDescription, Is.EqualTo(expectedDescription));

}

[Test]

public void LookAtGemInNoBag()

{

Player player = new Player("Tan", "A player");

LookCommand LookCommand = new LookCommand();

string expectedDescription = "I can't find the backpack";

string testDescription = LookCommand.Execute(player, new string[] { "look", "at", "gem", "in", "backpack" });

Assert.That(testDescription, Is.EqualTo(expectedDescription));

}

[Test]

public void LookAtNoGemInBag()

{

Player player = new Player("Tan", "A player");

Bag backpack = new Bag(new string[] { "backpack" }, "backpack", "a backpack");

player.Inventory.Put(backpack);

LookCommand LookCommand = new LookCommand();

string expectedDescription = "I can't find the gem in the backpack";

string testDescription = LookCommand.Execute(player, new string[] { "look", "at", "gem", "in", "backpack" });

Assert.That(testDescription, Is.EqualTo(expectedDescription));

}

[Test]

public void InvalidLookCommand()

{

Player player = new Player("Tan", "A player");

LookCommand LookCommand = new LookCommand();

string expectedDescription = "I don't know how to look like that";

//only 2 arguments

string testDescription = LookCommand.Execute(player, new string[] { "look", "at" });

Assert.That(testDescription, Is.EqualTo(expectedDescription));

//4 arguments

string testDescription2 = LookCommand.Execute(player, new string[] { "look", "at", "gem", "in" });

Assert.That(testDescription2, Is.EqualTo(expectedDescription));

//5 arguments but the 4th argument is not "in"

string testDescription3 = LookCommand.Execute(player, new string[] { "look", "at", "a", "at", "b" });

string expectedDescription2 = "What do you want to look in?";

Assert.That(testDescription3, Is.EqualTo(expectedDescription2));

//5 arguments but the 2nd argument is not "at"

string testDescription4 = LookCommand.Execute(player, new string[] { "look", "in", "a", "in", "b" });

string expectedDescription3 = "What do you want to look at?";

Assert.That(testDescription4, Is.EqualTo(expectedDescription3));

}

**Test Results**

**A screenshot of a computer program

Description automatically generated**