CommandProcessor.cs

```
using static PowerArgs.Ansi.Cursor;

namespace Iteration1
{
    public class CommandProcessor : Command
    {
        List<Command> _commands;

        public CommandProcessor() : base(new string[] { "command" })
        {
            _commands = new List<Command>();
            _commands.Add(new LookCommand());
            _commands.Add(new MoveCommand());
        }

        public override string Execute(Player p, string[] text)
        {
            string[] array = text[0].Split(' ');
            foreach (Command command in _commands)
            {
                if (command.AreYou(array[0]))
                {
                    return command.Execute(p, new string[] { text[0] });
                }
            }
            return "Wrong command!!!";
        }
    }
}

```

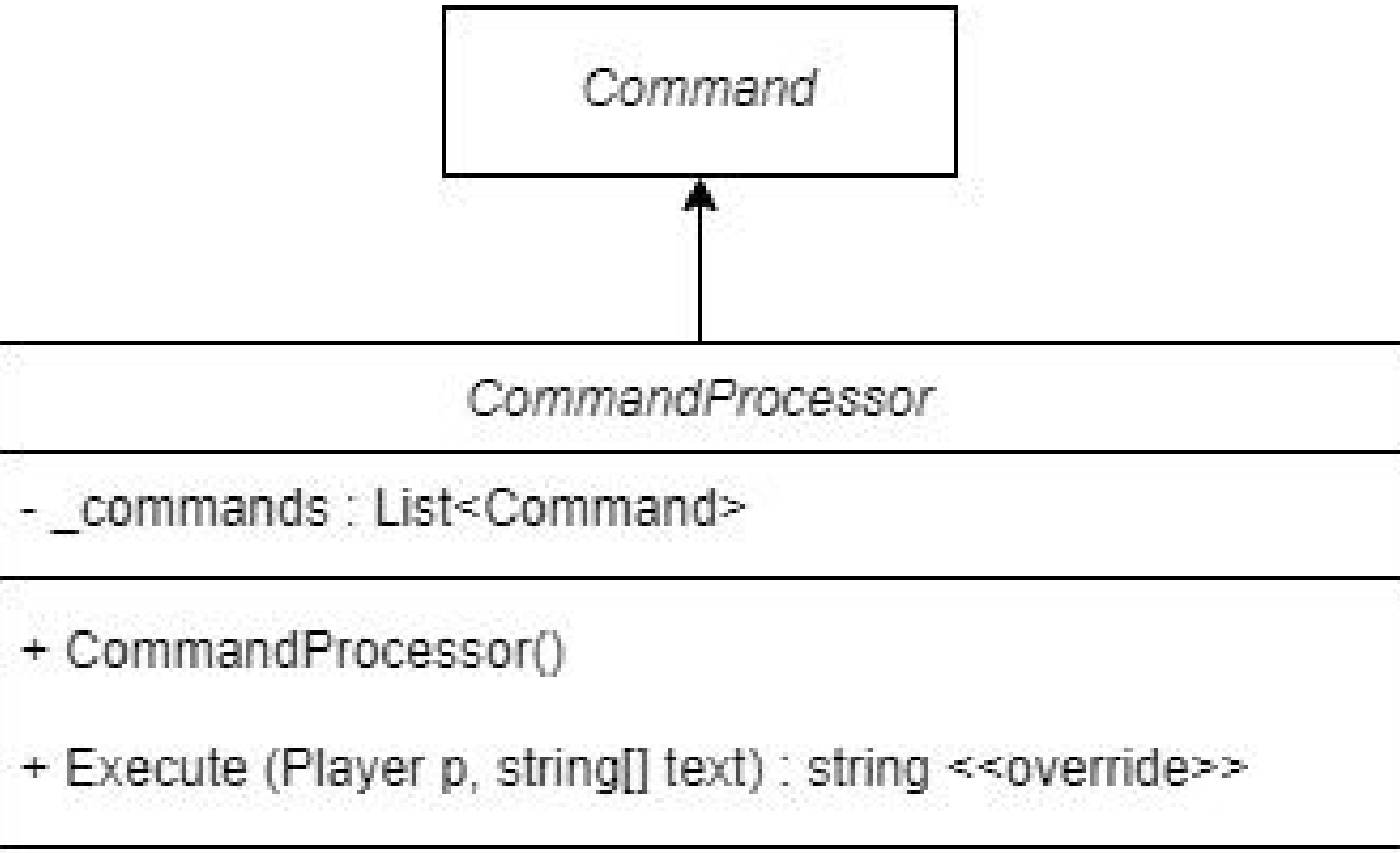
Program.cs

```csharp
using System.Runtime.InteropServices;

namespace Iteration1
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Console.Write("Player name: ");
            string name = Console.ReadLine();
            Console.Write("Player description: ");
            string desc_ = Console.ReadLine();
            Player player = new Player(name, desc_);

            Item sword = new Item(new string[] { "sword" }, "sword", "Short range weapon!");
            Item ak47 = new Item(new string[] { "ak47" }, "ak47", "Average range weapon with high damage!");

            player.Inventory.Put(sword);
            player.Inventory.Put(ak47);

            Item grenade = new Item(new string[] { "grenade" }, "grenade", "Extreme damage and short range weapon!");
            Bag bag1 = new Bag(new string[] { "bag1" }, "bag1", "");
            bag1.Inventory.Put(grenade);
            player.Inventory.Put(bag1);

            Location location = new Location(new string[] { "military base" }, "military base", "large area");
            Path _northpath = new Path(new string[] { "north" }, "hospital", "this is a hospital");

            location.AddPath(_northpath);
            player.CurrentLocation = location;

            while (true)
            {
                Console.Write("Command -> ");
                string command = Console.ReadLine();
                CommandProcessor selected_command = new CommandProcessor();
                string message = selected_command.Execute(player, new string[] {command});
                Console.WriteLine(message);
            }
        }
    }
}
```

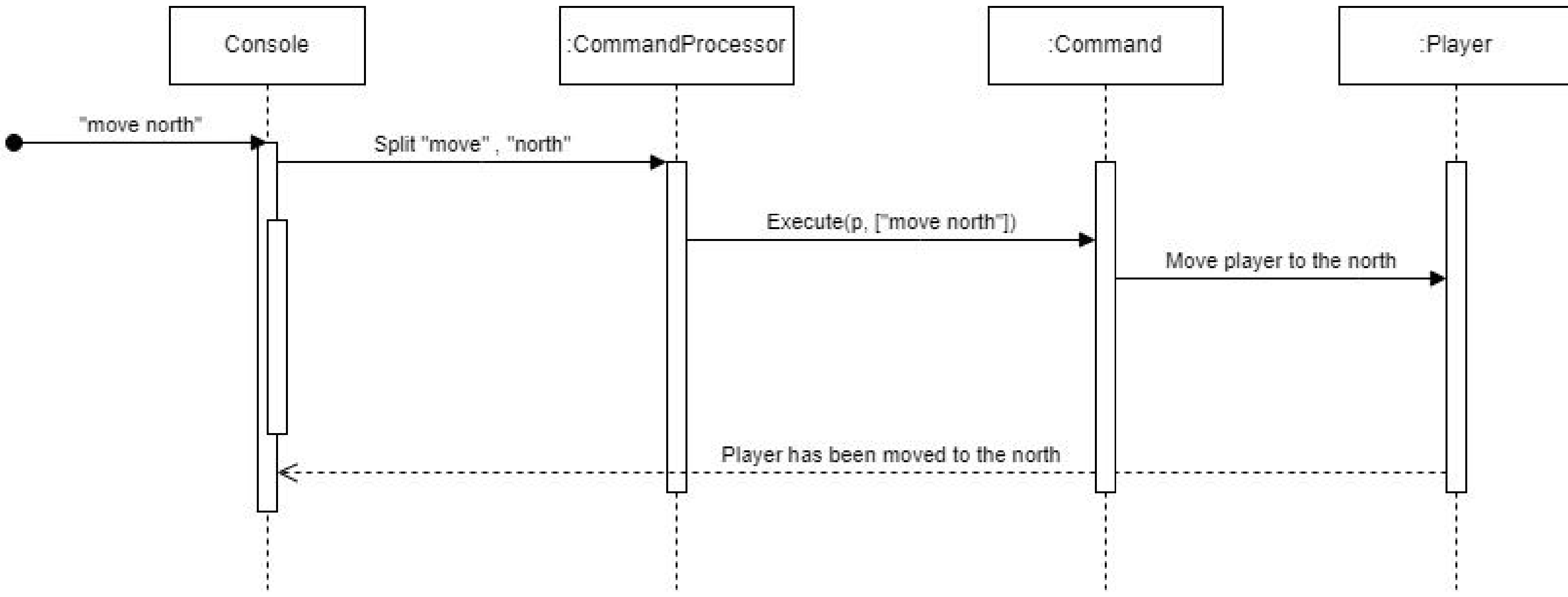
CommandProcessorUnitTest.cs

```
using Iteration1;
using static PowerArgs.Ansi.Cursor;
using Path = Iteration1.Path;

namespace CommandProcessorUnitTest
{
    public class Tests
    {
        private CommandProcessor _command;
        private Player _player;
        private Location _location;
        private Item _sword;
        private Item _ak47;
        private Item _grenade;
        private Path _northpath;

        [SetUp]
        public void Setup()
        {
            _command = new CommandProcessor();
            _player = new Player("Chien", "A boy with high curiosity");
            _location = new Location(new string[] { "military base" },
              "military base", "large area");
            _sword = new Item(new string[] { "sword", "melee" }, "sword",
              "Short range weapon");
            _ak47 = new Item(new string[] { "ak47" }, "ak47", "Long range
              weapon");
            _grenade = new Item(new string[] { "grenade" }, "grenade",
              "Very high damage weapon!");
            _location.Inventory.Put(_sword);
            _location.Inventory.Put(_ak47);
            _player.CurrentLocation = _location;
            _northpath = new Path(new string[] { "north" }, "hospital",
              "this is a hospital");
            _location.AddPath(_northpath);
        }

        [Test]
        public void TestMoveCommand()
        {
            Assert.That(_command.Execute(_player, ["move north"]),
              Is.EqualTo("You have moved to hospital\n"));
            Assert.Pass();
        }

        [Test]
        public void TestLookCommand()
        {
            Assert.That(_command.Execute(_player, ["look at me"]),
                Is.EqualTo(_player.FullDescription + "\n"));
            Assert.Pass();
        }

        [Test]
        public void TestWrongCommand()
        {
            Assert.That((_command.Execute(_player, new string[] { "look
              around" })), Is.EqualTo("What do you want to look at?\n"));
            Assert.That(_command.Execute(_player, new string[] { "run
              north" }), Is.EqualTo("Wrong command!!!"));
            Assert.That(_command.Execute(_player, new string[]
              { "hello" }), Is.EqualTo("Wrong command!!!"));
            Assert.Pass();
        }
    }
}
```

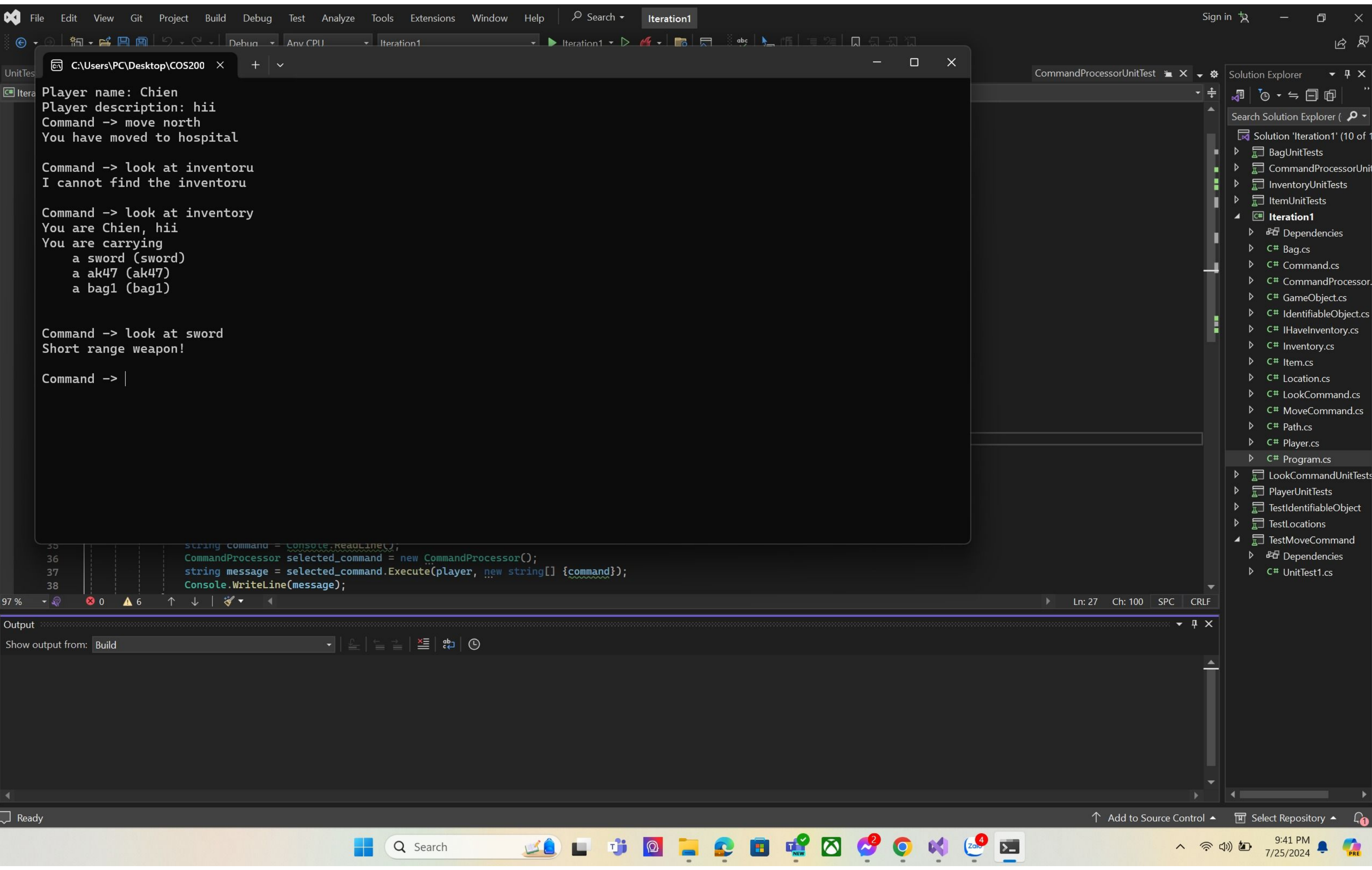
UML Class Diagram


Command
CommandProcessor
- _commands : List<Command>
+ CommandProcessor()
+ Execute (Player p, string[] text) : string <<override>>

UML Sequence Diagram



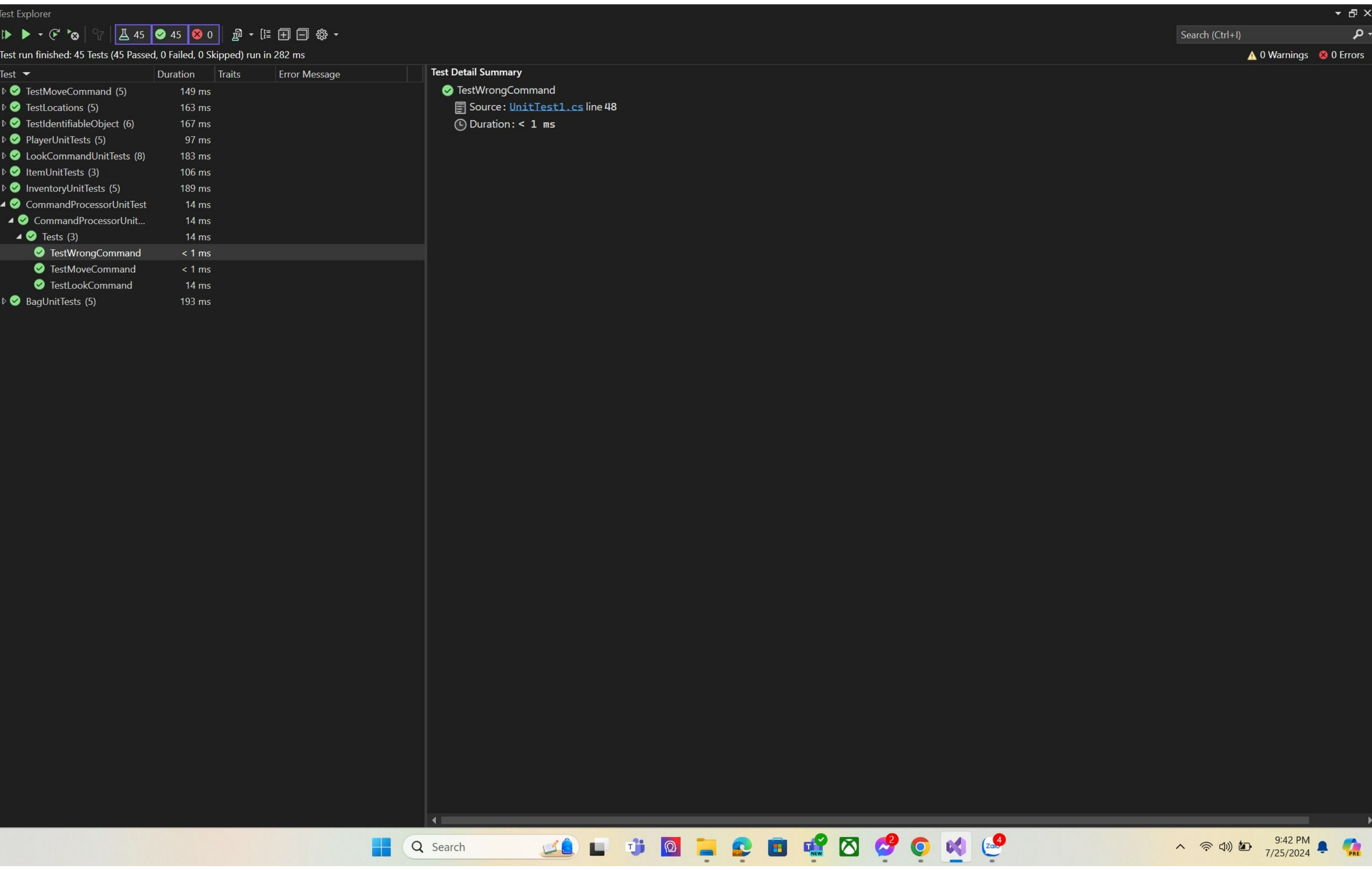
Program Running



```
Player name: Chien
Player description: hii
Command -> move north
You have moved to hospital

Command -> look at inventoru
I cannot find the inventoru

Command -> look at inventory
You are Chien, hii
You are carrying
    a sword (sword)
    a ak47 (ak47)
    a bag1 (bag1)


Command -> look at sword
Short range weapon!

Command ->
```

```
string command = Console.ReadLine();
CommandProcessor selected_command = new CommandProcessor();
string message = selected_command.Execute(player, new string[] {command});
Console.WriteLine(message);
```

UnitTest Passing



Test Explorer

45 45 0

Test run finished: 45 Tests (45 Passed, 0 Failed, 0 Skipped) run in 282 ms

0 Warnings 0 Errors

Test	Duration	Traits	Error Message
TestMoveCommand (5)	149 ms		
TestLocations (5)	163 ms		
TestIdentifiableObject (6)	167 ms		
PlayerUnitTests (5)	97 ms		
LookCommandUnitTests (8)	183 ms		
ItemUnitTests (3)	106 ms		
InventoryUnitTests (5)	189 ms		
CommandProcessorUnitTest	14 ms		
CommandProcessorUnit...	14 ms		
Tests (3)	14 ms		
TestWrongCommand	< 1 ms		
TestMoveCommand	< 1 ms		
TestLookCommand	14 ms		
BagUnitTests (5)	193 ms		

Test Detail Summary

TestWrongCommand

Source: UnitTest1.cs line 48

Duration: < 1 ms