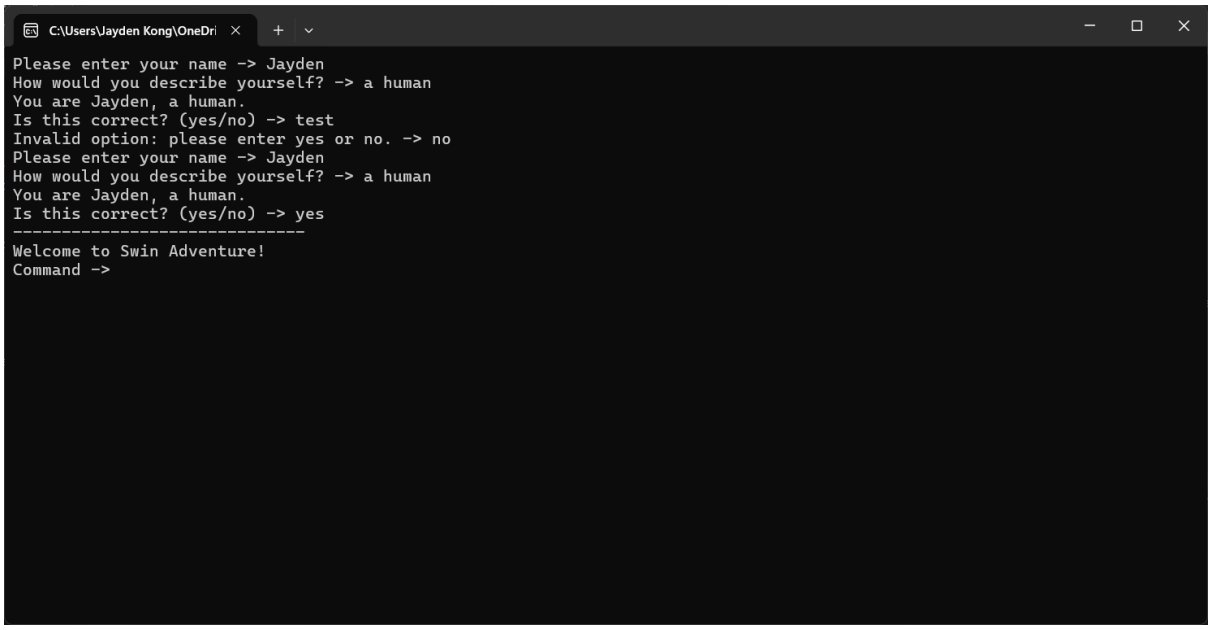
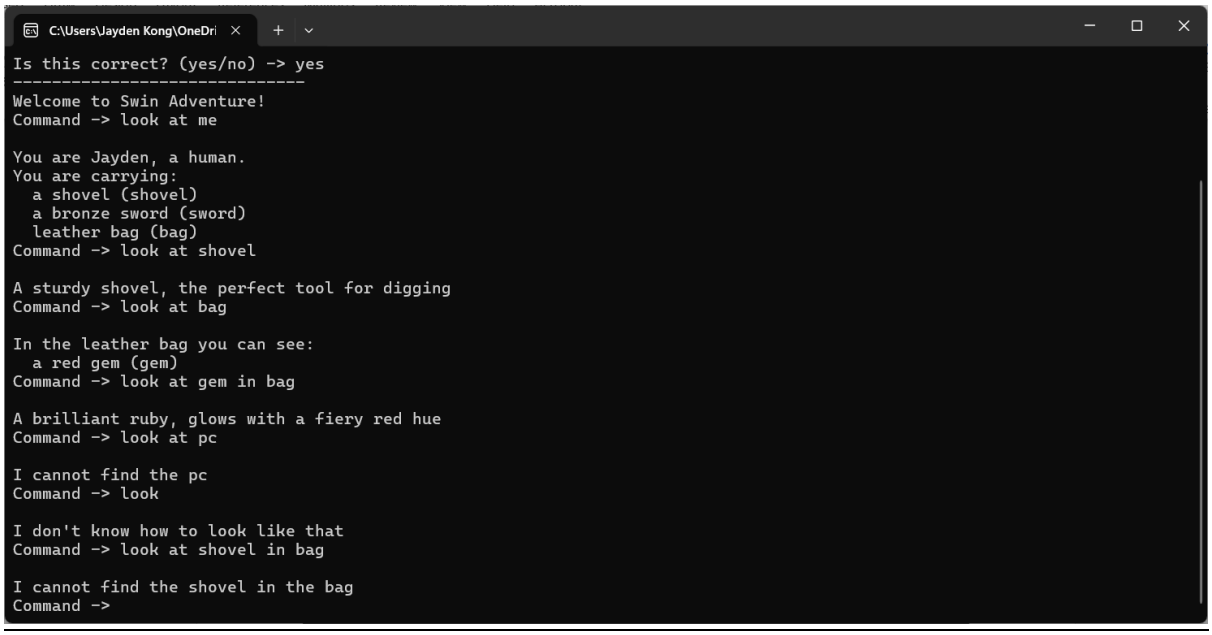
7.1P - Case Study - Iteration 5 - Tying it Together

Jayden Kong, 104547242

Player Creation:



Looping look commands:



```
1  namespace SwinAdventure
2  {
3      internal class Program
4      {
5          static void Main(string[] args)
6          {
7              Player? player = null;
8              Command lookCommand = new LookCommand();
9              while (player == null)
10             {
11                 Console.Write("Please enter your name -> ");
12                 string? playerName = Console.ReadLine();
13                 Console.Write("How would you describe yourself? -> ");
14                 string playerDescription = Console.ReadLine();
15                 Console.Write("You are {0}, {1}.\nIs this correct? (yes/no)
                       -> ", playerName, playerDescription);
16                 bool confirmationMenuLoop = true;
17                 while (confirmationMenuLoop)
18                 {
19                     string? decision = Console.ReadLine().ToLower();
20                     switch (decision)
21                     {
22                         case "yes":
23                             player = new Player(playerName,
                           playerDescription);
24                             confirmationMenuLoop = false;
25                             break;
26                         case "no":
27                             confirmationMenuLoop = false;
28                             break;
29                         default:
30                             Console.Write("Invalid option: please enter yes
                           or no. -> ");
31                             break;
32                     }
33                 }
34             }
35
36             Console.WriteLine("------------------------------");
37             Console.WriteLine("Welcome to Swin Adventure!");
38
39             Item shovel = new Item(new string[] { "shovel", "spade" }, "a
                 shovel", "A sturdy shovel, the perfect tool for digging");
40             Item bronzeSword = new Item(new string[] { "sword" }, "a bronze
                  sword", "A short sword forged from bronze");
41             Item ruby = new Item(new string[] { "gem", "ruby" }, "a red
                 gem", "A brilliant ruby, glows with a fiery red hue");
42             Bag bag = new Bag(new string[] { "bag" }, "leather bag",
                 "Crafted from supple brown leather, this small bag is perfect
```

```
                for carrying items");
            bag.Inventory.Put(ruby);
            player.Inventory.Put(shovel);
            player.Inventory.Put(bronzeSword);
            player.Inventory.Put(bag);

            bool gameLoop = true;
            while (gameLoop)
            {
                Console.Write("Command -> ");
                string? playerInput = Console.ReadLine();
                string[] inputToPass = playerInput.Split(new char[] {  },
                  StringSplitOptions.RemoveEmptyEntries);           //
                  Temporary code for passing in things to the look command
                  until iteration 8
                Console.WriteLine("");
                Console.WriteLine(lookCommand.Execute(player,
                  inputToPass));
            }
        }
    }
}

```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public class IdentifiableObject
10     {
11         private List<string> _identifiers;
12
13         public string FirstID
14         {
15             get
16             {
17                 if (_identifiers.Count > 0)
18                 {
19                     return _identifiers[0];
20                 }
21                 return "";
22             }
23         }
24
25         public IdentifiableObject(string[] idents)
26         {
27             _identifiers = new List<string>();
28             for (int i = 0; i < idents.Length; i++)
29             {
30                 _identifiers.Add(idents[i].ToLower());
31             }
32         }
33
34         public bool AreYou(string id)
35         {
36             bool result = false;
37
38             foreach (string ident in _identifiers)
39             {
40                 if (ident == id.ToLower())
41                 {
42                     result = true;
43                     break;
44                 }
45             }
46
47             return result;
48         }
49
```

```
50          public void AddIdentifier(string id)
51          {
52              _identifiers.Add(id.ToLower());
53          }
54      }
55  }
56
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel.Design;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace SwinAdventure
9  {
10     public abstract class GameObject : IdentifiableObject
11     {
12         private string _description;
13         private string _name;
14
15         public string Name
16         {
17             get
18             {
19                 return _name;
20             }
21         }
22
23         public string ShortDescription
24         {
25             get
26             {
27                 return string.Format("{0} ({1})", Name, base.FirstID);
28             }
29         }
30
31         public virtual string FullDescription
32         {
33             get
34             {
35                 return _description;
36             }
37         }
38
39         public GameObject(string[] ids, string name, string desc) : base
             (ids)
40         {
41             _name = name;
42             _description = desc;
43         }
44     }
45 }
46
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public class Player : GameObject, IHaveInventory
10     {
11         private Inventory _inventory;
12
13         public override string FullDescription
14         {
15             get
16             {
17                 return string.Format("You are {0}, {1}.\nYou are carrying:
                       {2}", Name, base.FullDescription, Inventory.ItemList);
18             }
19         }
20
21         public Inventory Inventory
22         {
23             get
24             {
25                 return _inventory;
26             }
27         }
28
29         public Player(string name, string desc) : base(new string[] {"me",
             "inventory"}, name, desc)
30         {
31             _inventory = new Inventory();
32         }
33
34         public GameObject? Locate(string id)
35         {
36             if (AreYou(id))
37             {
38                 return this;
39             }
40             return Inventory.Fetch(id);
41         }
42     }
43 }
44
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public class Item : GameObject
10     {
11         public Item(string[] idents, string name, string desc) : base
             (idents, name, desc) { }
12     }
13 }
14
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public class Inventory
10     {
11         private List<Item> _items;
12
13         public string ItemList
14         {
15             get
16             {
17                 string itemList = "";
18                 foreach (Item item in _items)
19                 {
20                     itemList += (string.Format("\n  {0}",
                       item.ShortDescription));
21                 }
22
23                 return itemList;
24             }
25         }
26
27         public Inventory()
28         {
29             _items = new List<Item>();
30         }
31
32         public bool HasItem(string id)
33         {
34             return Fetch(id) != null;
35         }
36
37         public void Put(Item itm)
38         {
39             _items.Add(itm);
40         }
41
42         public Item? Take(string id)
43         {
44             foreach (Item item in _items)
45             {
46                 if (item.AreYou(id))
47                 {
48                     _items.Remove(item);
```

```
49                      return item;
50                  }
51              }
52              return null;
53          }
54
55          public Item? Fetch(string id)
56          {
57              foreach (Item item in _items)
58              {
59                  if (item.AreYou(id))
60                  {
61                      return item;
62                  }
63              }
64              return null;
65          }
66
67      }
68  }
69
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public class Bag : Item, IHaveInventory
10     {
11         private Inventory _inventory;
12
13         public override string FullDescription
14         {
15             get
16             {
17                 return string.Format("In the {0} you can see: {1}", Name,
                       Inventory.ItemList);
18             }
19         }
20
21         public Inventory Inventory
22         {
23             get
24             {
25                 return _inventory;
26             }
27         }
28
29         public Bag(string[] ids, string name, string desc): base(ids, name,
               desc)
30         {
31             _inventory = new Inventory();
32         }
33
34         public GameObject? Locate(string id)
35         {
36             if (AreYou(id))
37             {
38                 return this;
39             }
40             return Inventory.Fetch(id);
41         }
42     }
43 }
44
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace SwinAdventure
9  {
10     public class LookCommand : Command
11     {
12         public LookCommand() : base(new string[] {"look"}) { }
13
14         public override string Execute(Player p, string[] text)
15         {
16             if (!(text.Length == 3 || text.Length == 5))
17             {
18                 return "I don't know how to look like that";
19             }
20
21             if (text[0] != "look")
22             {
23                 return "Error in look input";
24             }
25
26             if (text[1] != "at")
27             {
28                 return "What do you want to look at?";
29             }
30
31             if (text.Length == 5 && text[3] != "in")
32             {
33                 return "What do you want to look in?";
34             }
35
36             if (text.Length == 3)
37             {
38                 string? itemDescription3 = LookAtIn(text[2], p);
39                 if (itemDescription3 == null)
40                 {
41                     return string.Format("I cannot find the {0}", text[2]);
42                 }
43                 return itemDescription3;
44             }
45
46             // By this point the 3 element look command is done
47             IHaveInventory? container = FetchContainer(p, text[4]);
48             if (container == null)
49             {
```

```
50                  return string.Format("I cannot find the {0}", text[4]);
51              }
52
53              string? itemDescription5 = LookAtIn(text[2], container);
54              if (itemDescription5 == null)
55              {
56                  return string.Format("I cannot find the {0} in the {1}",
                      text[2], text[4]);
57              }
58              return itemDescription5;
59          }
60
61          private IHaveInventory? FetchContainer(Player p, string
              containerId)
62          {
63              IHaveInventory? container = p.Locate(containerId) as
                  IHaveInventory;
64              return container;
65          }
66
67          private string? LookAtIn(string thingId, IHaveInventory container)
68          {
69              GameObject? item = container.Locate(thingId);
70              if (item == null)
71              {
72                  return null;
73              }
74              return item.FullDescription;
75          }
76      }
77  }
78
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public interface IHaveInventory
10     {
11         public string Name { get; }
12
13         public GameObject? Locate(string id);
14     }
15 }
16
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public abstract class Command : IdentifiableObject
10     {
11         public Command(string[] ids) : base(ids) { }
12
13         public abstract string Execute(Player p, string[] text);
14     }
15 }
16
```