Project Report:

Critical Thinking 1 – RPG Bag

Alejandro Ricciardi

Colorado State University Global

CSC400: Data Structures and Algorithms

Professor: Hubert Pensado

August 18, 2024

Project Report:

Critical Thinking 1 – RPG Bag

This documentation is part of the Critical Thinking 1 Assignment from CSC400: Data Structures and Algorithms at Colorado State University Global. This Project Report is an overview of the program's functionality and testing scenarios including console output screenshots. The program is coded in Java JDK-22 and named Critical Thinking 1 (RPG Bag).

The Assignment Direction:

Java Bag Data Structure

In this assignment, you will implement a Java bag data structure, also known as a multiset. A bag is a collection of elements that allows duplicates and does not enforce any particular order. Your task is to design and implement a bag class in Java that supports basic operations such as adding elements, removing elements, checking if an element exists, and counting the occurrences of an element.

1. Design a Java class called `Bag` that implements the bag data structure.
2. The `Bag` class should have the following methods:
    - `void add(T item)`: This method should add an item of type T to the bag.
    - `void remove(T item)`: This method should remove one occurrence of the item from the bag, if it exists.
    - `boolean contains(T item)`: This method should return true if the item exists in the bag; otherwise, it should return false.
3. Write a Java program that demonstrates the usage of the `Bag` class. Your program should perform the following operations:
4. Comment your code appropriately to explain the functionality of each method.
    - Create an instance of the `Bag` class.
    - Add several elements to the bag, including duplicates.
    - Print the bag contents.
    - Test the `contains` method for a few elements.
    - Test the `count` method for a few elements.
    - Remove an element from the bag.
    - Print the bag contents again
    - Test the `contains` method for the removed element.
    - Test the `count` method for the removed element.

Submit your completed assignment as a .java source code file.

⚠ **My notes:**

- The program implements an Item class that acts as the base class for the classes Potion, Armor, and Weapon.
- The class Bag implements the Iterable interface and uses a linked list, [element | next] -> [element | next] -> [element | next] -> null, to store elements.
- A popular implementation of the Bag ADT is to use a HashMap. Although a HashMap does not allow duplicate entries, it can store a single entry for each element along with its current count. This would eliminate the need to iterate through the entire Bag to count

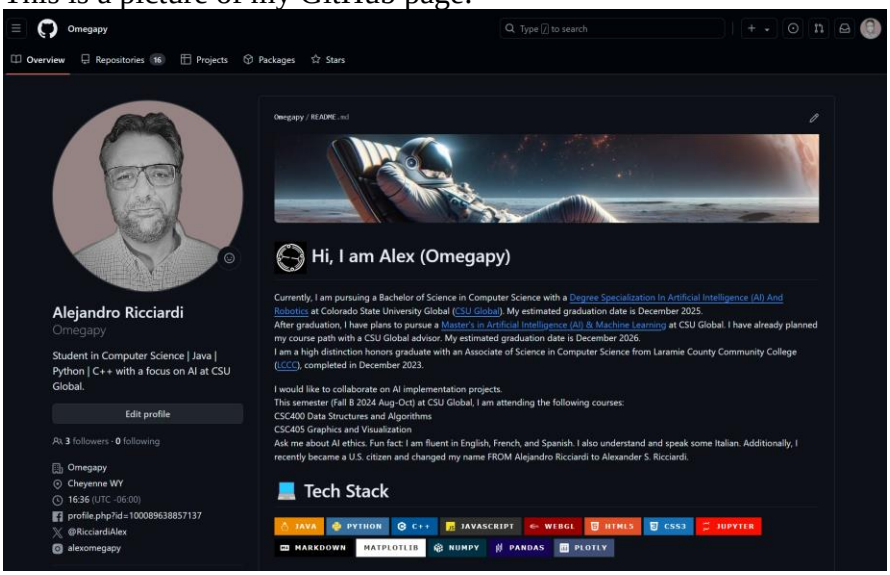
specific elements. However, for this assignment, I chose to use a linked list structure to show a more traditional approach to implementing a Bag ADT.
- In addition to the required functionalities, the program includes an extra feature that allows changing the price of an item object.
- The program source code can be found in the following files:
  - Item.java
  - Armor.java
  - Weapon.java
  - Potion.java
  - Bag.java
  - Main.java

My Program Description:

The program is an implementation of a Bag Abstract Data Structure (Bag ADT) using a Linked list structure.
[element | next] -> [element | next] -> [element | next] -> null.
The Bag class represents the inventory of an RPG video game player.
The Bag allows for the storage and management of game items such as Potions, Armor, and Weapons.
The Bag ADT is implemented as a generic class that can store any element object type.

Git Repository

This is a picture of my GitHub page:

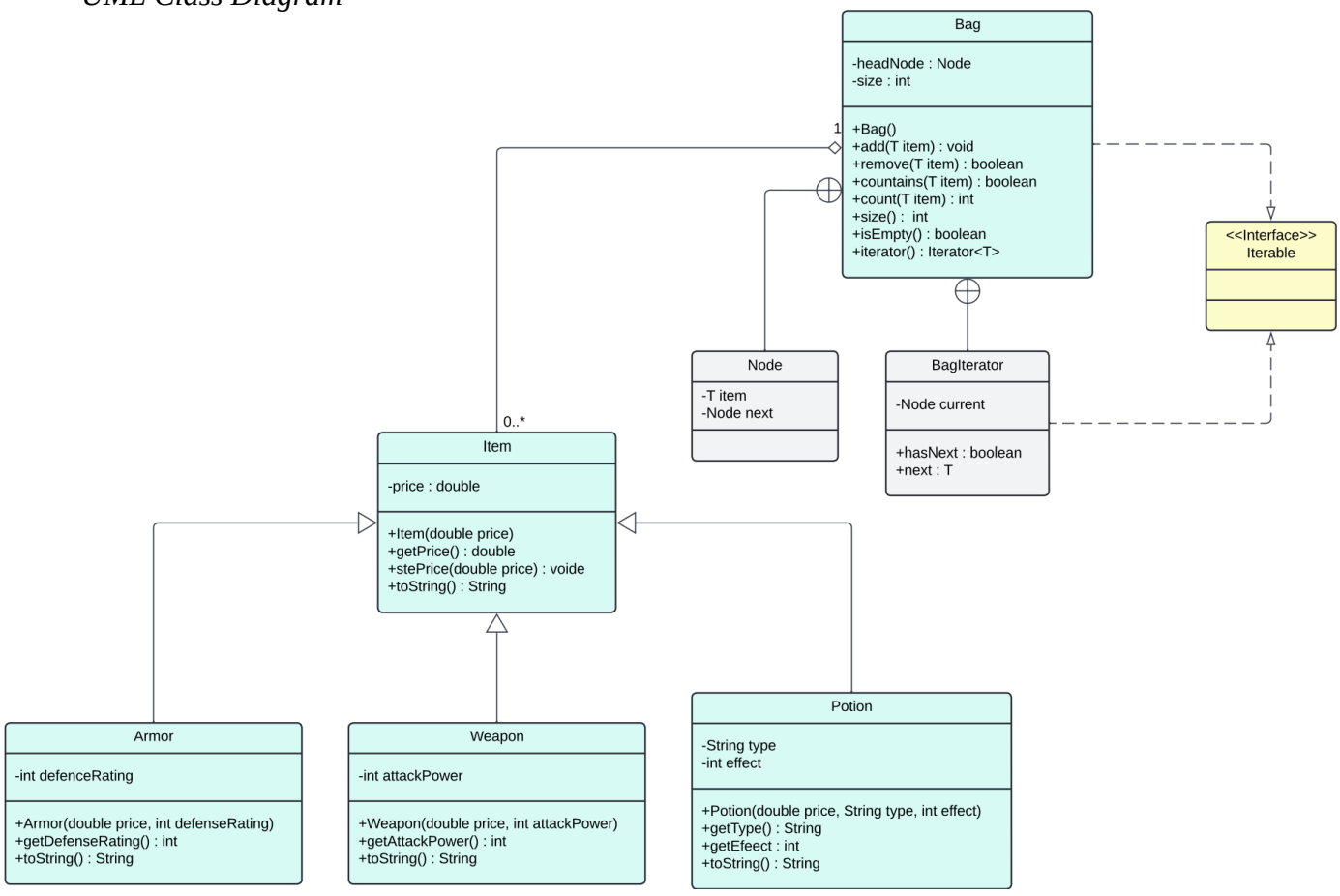


I use GitHub as my Distributed Version Control System (DVCS), the following is a link to my GitHub, Omegapy.
My GitHub repository that is used to store this assignment is named My-Academics-Portfolio and the link to this specific assignment is: https://github.com/Omegapy/My-Academics-Portfolio/tree/main/Data-Structures-and-Algorithms-CSC400/Critical-Thinking-1

Classes Description:

- **Armor Class**
  It extends the Item Class. It represents an armor item in the RPG game.
- **Weapon Class**
  It extends the Item Class. It represents a weapon item in the RPG game.
- **Potion Class**
  It extends the Item Class. It represents a potion item in the RPG game.
- **Weapon Class**
  This Class is the base class for all item types, such as Potion, Armor, and Weapon. It represents a generic item in the RPG game.
- **Weapon Class**
  This Class is the base class for all item types, such as Potion, Armor, and Weapon. It represents a generic item in the RPG game.
  Bag
  A Bag ADT class. This class implements a bag using a list structure,
  [element | next] -> [element | next] -> [element | next] -> null, to store elements.
- **Main**
  Runs tests on the functionality of the Bag data structure.

Figure 1
UML Class Diagram



Screenshot

Program Functionality

Figure 3
Initial Bag Inventory

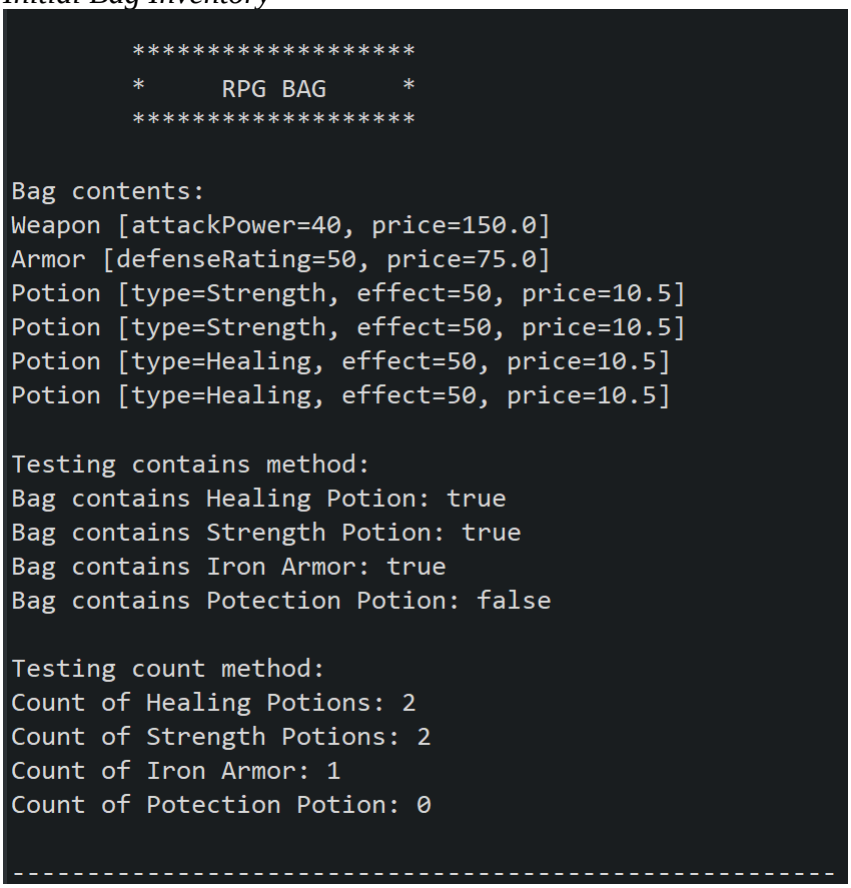
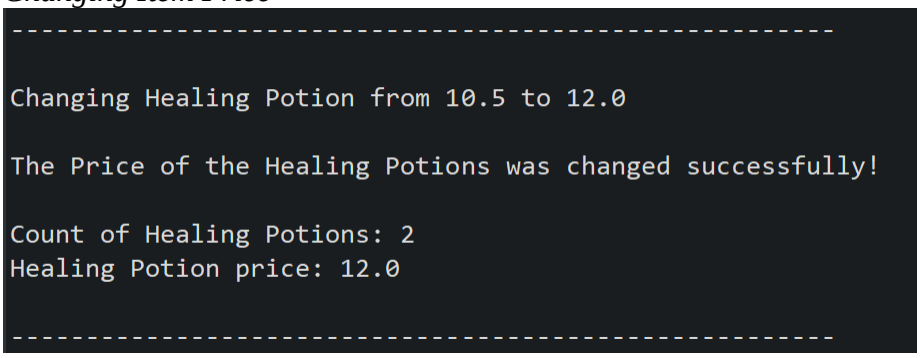


Figure 2
Changing Item Price



Continue next page

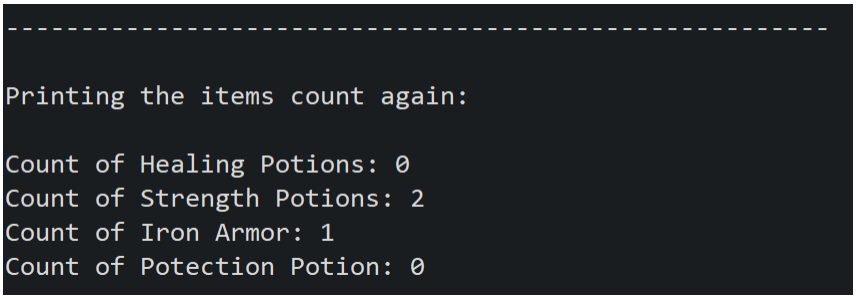
Figure 3
Deleting Items

```
-------------------------------------------------------

Deleting Healing Potion!

A Healing Potion was removed successfully!
Count of Healing Potions after removal: 1

Bag contents:
Weapon [attackPower=40, price=150.0]
Armor [defenseRating=50, price=75.0]
Potion [type=Strength, effect=50, price=10.5]
Potion [type=Strength, effect=50, price=10.5]
Potion [type=Healing, effect=50, price=12.0]

-------------------------------------------------------

Deleting Healing Potion!

Another Healing Potion was removed successfully!
Count of Healing Potions after removal: 0

Bag contents:
Weapon [attackPower=40, price=150.0]
Armor [defenseRating=50, price=75.0]
Potion [type=Strength, effect=50, price=10.5]
Potion [type=Strength, effect=50, price=10.5]

-------------------------------------------------------

Deleting Healing Potion!

The Bag does not contain any Healing Potion!
Count of Healing Potions: 0

Bag contents:
Weapon [attackPower=40, price=150.0]
Armor [defenseRating=50, price=75.0]
Potion [type=Strength, effect=50, price=10.5]
Potion [type=Strength, effect=50, price=10.5]

-------------------------------------------------------
```

Continue next page

Figure 4

Final Count Check



As shown in Figure 2 through Figure 4 the program runs without any issues displaying the correct outputs as expected.