CSCI 345 - Object Oriented Design

Assignment 01 Part E

Comparing and Cloning Objects

Overview

This assignment is designed to give you help understanding how to compare and clone objects . You can work with one other student on this assignment, or you can elect to complete the assignment on your own.

Program Specification

Mr. Pumphrey received further feedback from Penelope with regards to the Backpack application (we all know by now that what Penelope wants, Penelope gets).

Penelope met with the lead developer, Chrysanthemum, and the Backpack application team members that Chrysanthemum leads. During the meeting, it was determined that the Backpack and Pocket classes should support object comparison and cloning.

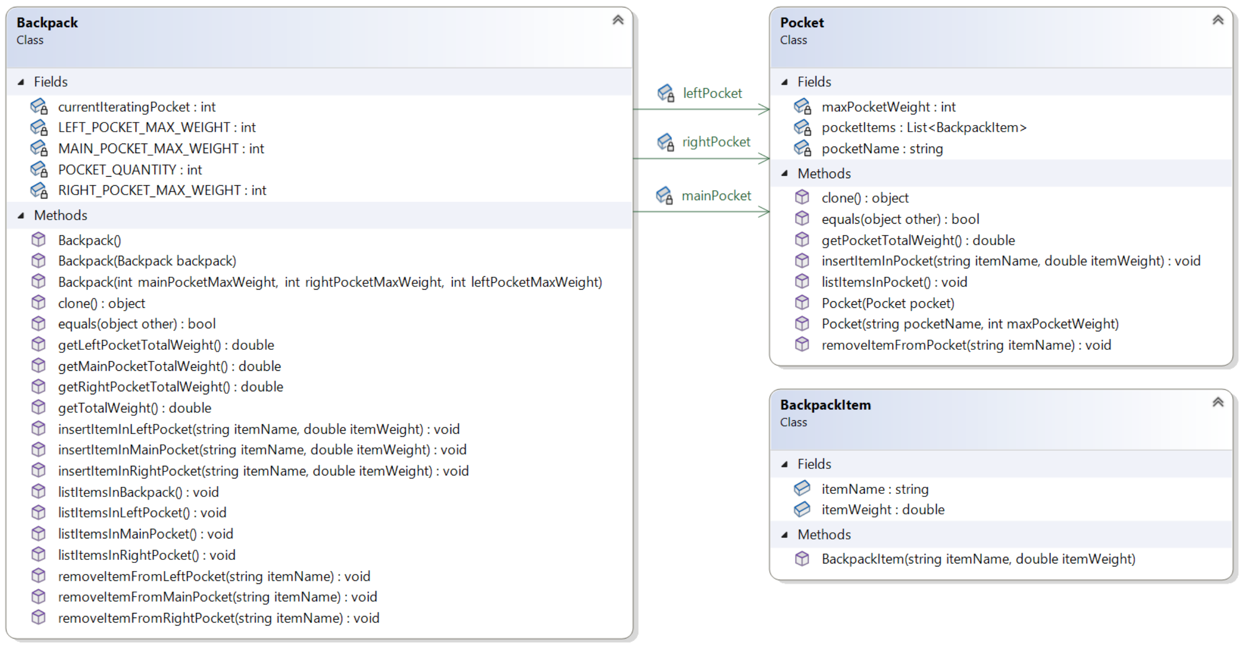
Mr. Pumphrey agreed that the backpack application should be refactored to support these new requirements, and he extended and updated the contract terms with Never Crash Software Services. A few other requirements were added also.

Here's a list of the new requirements:

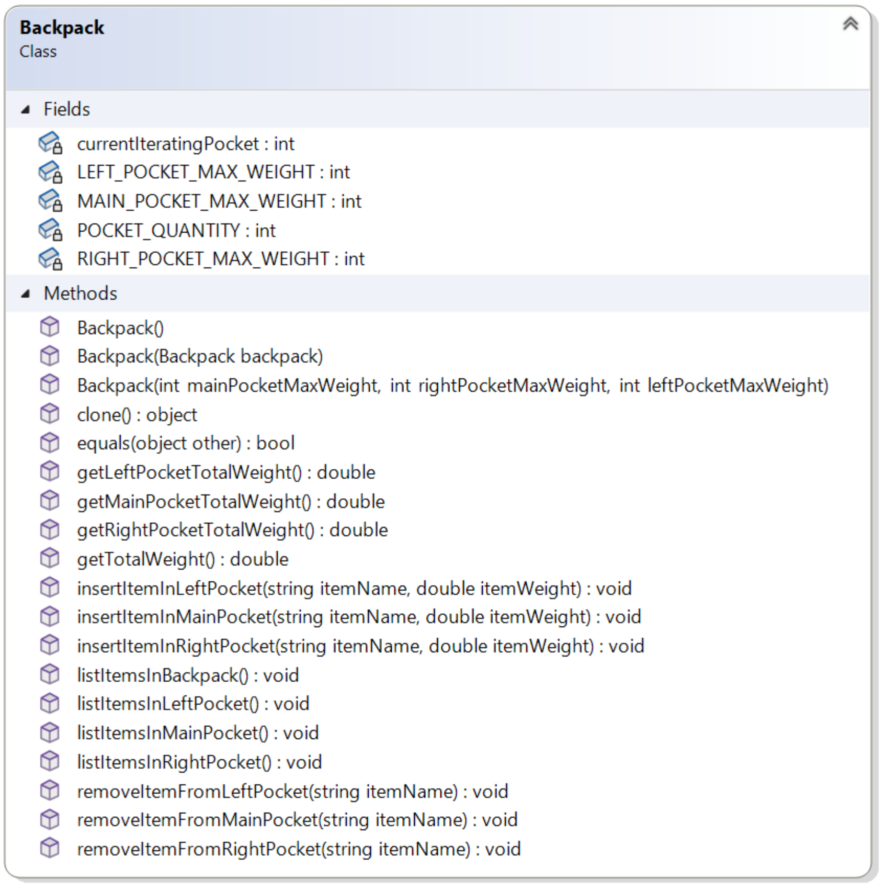
- Override the base class Object's equals method for both the Backpack and Pocket classes.
  - NOTE: To be considered equal, the items in the two pockets being compared do not have to be in the same order.
- Override the base class Object's clone method for both the Backpack and Pocket classes. A deep clone is expected for both Backpack's override of clone and for Pocket's override of clone.
- Add a copy constructor for both the Backpack and Pocket classes.
- Change the data type of Pocket.pocketItems from the concrete class ArrayList to the interface List. This would allow future versions to use any type of class that implements the List interface, such as the LinkedList concrete class.

The UML diagrams for the Backpack and Pocket classes are shown below. A partially completed Backpack class, Pocket class, and test program (Program.java) are uploaded. You can download the files from Canvas->Files-> Assignments-> Assignment_01_E.

UML Class Diagram



Backpack
Class
Fields
currentIteratingPocket : int
LEFT_POCKET_MAX_WEIGHT : int
MAIN_POCKET_MAX_WEIGHT : int
POCKET_QUANTITY : int
RIGHT_POCKET_MAX_WEIGHT : int
Methods
Backpack()
Backpack(Backpack backpack)
Backpack(int mainPocketMaxWeight, int rightPocketMaxWeight, int leftPocketMaxWeight)
clone() : object
equals(object other) : bool
getLeftPocketTotalWeight() : double
getMainPocketTotalWeight() : double
getRightPocketTotalWeight() : double
getTotalWeight() : double
insertItemInLeftPocket(string itemName, double itemWeight) : void
insertItemInMainPocket(string itemName, double itemWeight) : void
insertItemInRightPocket(string itemName, double itemWeight) : void
listItemsInBackpack() : void
listItemsInLeftPocket() : void
listItemsInMainPocket() : void
listItemsInRightPocket() : void
removeItemFromLeftPocket(string itemName) : void
removeItemFromMainPocket(string itemName) : void
removeItemFromRightPocket(string itemName) : void
leftPocket
rightPocket
mainPocket
Pocket
Class
Fields
maxPocketWeight : int
pocketItems : List<BackpackItem>
pocketName : string
Methods
clone() : object
equals(object other) : bool
getPocketTotalWeight() : double
insertItemInPocket(string itemName, double itemWeight) : void
listItemsInPocket() : void
Pocket(Pocket pocket)
Pocket(string pocketName, int maxPocketWeight)
removeItemFromPocket(string itemName) : void
BackpackItem
Class
Fields
itemName : string
itemWeight : double
Methods
BackpackItem(string itemName, double itemWeight)



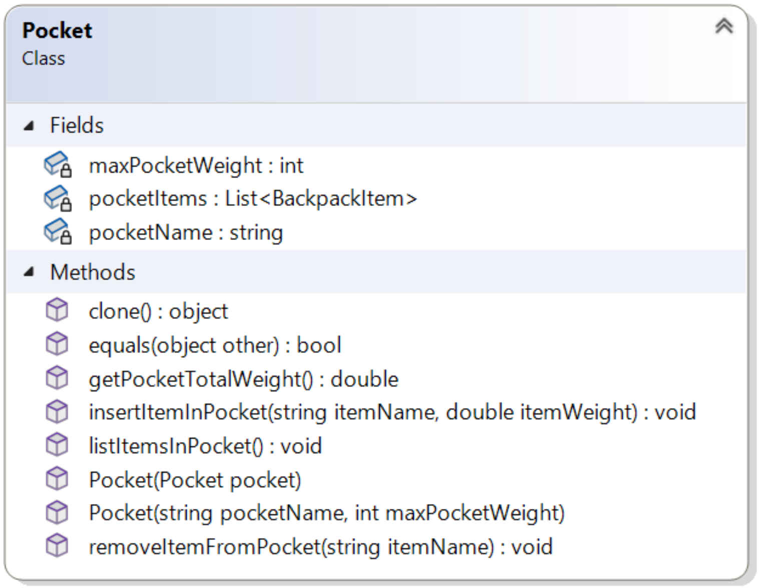
Backpack
Class

Fields

- currentIteratingPocket : int
- LEFT_POCKET_MAX_WEIGHT : int
- MAIN_POCKET_MAX_WEIGHT : int
- POCKET_QUANTITY : int
- RIGHT_POCKET_MAX_WEIGHT : int

Methods

- Backpack()
- Backpack(Backpack backpack)
- Backpack(int mainPocketMaxWeight, int rightPocketMaxWeight, int leftPocketMaxWeight)
- clone() : object
- equals(object other) : bool
- getLeftPocketTotalWeight() : double
- getMainPocketTotalWeight() : double
- getRightPocketTotalWeight() : double
- getTotalWeight() : double
- insertItemInLeftPocket(string itemName, double itemWeight) : void
- insertItemInMainPocket(string itemName, double itemWeight) : void
- insertItemInRightPocket(string itemName, double itemWeight) : void
- listItemsInBackpack() : void
- listItemsInLeftPocket() : void
- listItemsInMainPocket() : void
- listItemsInRightPocket() : void
- removeItemFromLeftPocket(string itemName) : void
- removeItemFromMainPocket(string itemName) : void
- removeItemFromRightPocket(string itemName) : void



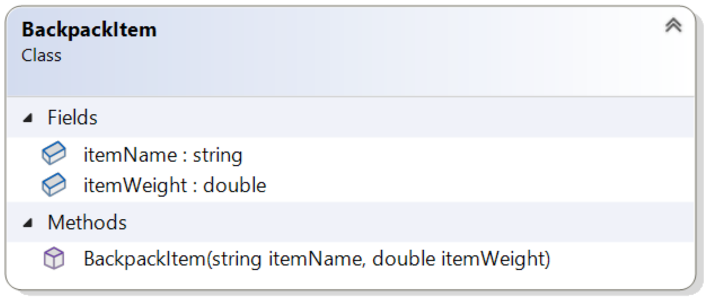
Pocket
Class

Fields

- maxPocketWeight : int
- pocketItems : List<BackpackItem>
- pocketName : string

Methods

- clone() : object
- equals(object other) : bool
- getPocketTotalWeight() : double
- insertItemInPocket(string itemName, double itemWeight) : void
- listItemsInPocket() : void
- Pocket(Pocket pocket)
- Pocket(string pocketName, int maxPocketWeight)
- removeItemFromPocket(string itemName) : void



Program Execution

Sample Program Execution

```
Running the Backpack application...

Creating backpack1 and backpack2 with the same items...
Assertion: backpack1 is equal to backpack2: true

Creating backpack3 by cloning backpack1...
Assertion: backpack3 is equal to backpack1: true
Assertion: backpack3 is equal to backpack2: true

Creating backpack4 by passing backpack1 to copy constructor...
Assertion: backpack4 is equal to backpack2: true
```

Comments

At the top of all source code files, add the following commenting:

```
/*

# Name:

# Date:

# Description:

*/
```

Submission

Upload the refactored Backpack.java and refactored Pocket.java files to Canvas. For this assignment, Canvas has been configured to permit only files that end with the .java file extension.