**CS-207: Programming II**
**Spring 2016**
**Northeastern Illinois University**
**Homework #9: Due 04/12/16 at 9:00 a.m.**
**Abstract Classes and Interfaces**

## Problem #1

You have been provided with four Java files for this problem. An abstract class named `Item.java`, two interfaces named `Sellable.java` and `Transportable.java`, and a test class named `TestProblem1.java`. Download the files from the NeededFiles.zip file and compile them (except for the test class).

Create a class named `Photograph` that does the following:

- `Photograph` should inherit from `Item` and implement the `Sellable` interface (this means that you must create/all abstract methods from both `Item` and `Sellable`.
- There should be two instance variables (encapsulated!): a `String` named `description` and a `double` named `price`.
- A constructor that takes three parameters: An `int` for `id`, a `String` for `name`, and a `double` for `price` (in that order exactly). It should set the `name` and `id` instance variables of the superclass, and the `price` instance variable of the `Photograph` class. Note that you are **not** initializing the `description` instance variable.
- Create a getter and setter method for the `description` instance variable. Note that the getter should be overriding/implementing the corresponding abstract method from the superclass.
- Create the `printDetails` method. It should print out the details in the following format:
  `ID: <id value here>`
  `Name: <name value here>`
  `Description: <description value here>`
- Create a getter method for the `price` instance variable. Note that the getter should be overriding/implementing the corresponding abstract method from the interface.
- Create the `calculateSalePrice` method. It should divide the method parameter by 100 and multiply the result by the `price` instance variable and return this value.
- Compile the `Photograph` class.

Create a class named `Album` that does the following:

- `Album` should inherit from `Item` and implement both the `Sellable` and `Transportable` interfaces (this means that you must create/all abstract methods from `Item`,`Sellable`, and `Transportable`.
- There should be two instance variables (encapsulated!): a `Photograph` array named `photos` and a `double` named `weight`. You do not know the length of the array, so you should not try to initialize it - just declare it.

- A constructor that takes four parameters: An `int` for `id`, a `String` for `name`, a `double` for `weight`, and a `Photograph` array for `photos` (in that order exactly). It should set the `name` and `id` instance variables of the superclass, and the `weight` and `photos` instance variable of the `Photograph` class. Note that you are **not** initializing the `description` instance variable.
- Create the `getDescription` method. The method should return a String that, when printed, will contain the word `"Weight"` followed by the value of the `weight` instance variable on its own line and then the name of each element in the `photos` instance variable, separated by a space, and the value of the `description` of each `Photogragh` element on its own line. Remember that `"\n"` creates a new line.
- Create the `printDetails` method. It should print out the details in the following format:
  ```
  ID: <id value here>
  Name:  <name value here>
  Price:  <price value here>
  Description:
  <description value here>
  ```
- Create the `getPrice` method. The price of an album should be created by summing up the prices of each of the `Photograph` elements in the `photos` array. If the `weight` of an album is greater than 10.0, add 5.00 to the price. Otherwise only add 2.50 to the price.
- Create the `calculateSalePrice` method. It should divide the method parameter by 100 and multiply the result by the `price` instance variable and return this value.
- Create the `isHazardous` method. It should return `false`.
- Create the `isFragile` method. It should return `true`.
- Create the `calculateShippingCost` method. It should start with a base shipping cost of 5.00. If `isFragile` is `true`, the cost should be increased by 5.00. If `isHazardous` is `true`, the cost should be increased by 5.00.
- Compile your class.

Finally, do the following:

- Run the `TestProblem1.java` file. If you created your classes, you will see the output below.
- Place your `Item.java` and `Album.java` files into the Homework9 folder to be submitted to D2L.

```
ID: 392840
Name: Leather
Price: 2.95
Description:
2.75 lbs
Winter.jpg Photos of cold cold Chicago
Summer.jpg null
SpringFall.jpg Photos of Spring and Fall

Shipping: 10.0
Shipping: 30.0
```

**A note on cheating/plagiarism:**
A plagiarism detector is used on all submitted code (across all sections) for homework assignments. If the plagiarism detector determines that 25% or more of your code for a particular assignment is plagiarized, you will receive a zero (i.e. an F) for that homework assignment, regardless of whether you cheated from someone or vice-versa. If you plagiarize half or more of the total homework assignments, you will receive a zero for the entire homework percentage.

**Submitting your assignment to D2L**

1. Make sure your name and assignment number are in the .java file(s) (as comments) and text file.

2. Place all your files in a folder and compress (i.e. .zip) the folder. Submit the .zip file to the Homework #9 folder on D2L. You should submit only one file - the .zip file. Do **NOT** upload multiple files.

3. Turn your homework in to D2L by the specified deadline (no late homework will be accepted - see syllabus for policies)