HW3: Java 1

Due Feb 3 by 11:59pm **Points** 30 **Submitting** a file upload

Note: If you're not familiar with programming with an IDE, a few tips. First off, be sure to make a new Java project for each problem (file->new->project, then select java project). Once you have a project, you can create as many classes as you need by right-clicking on the folder containing your Main class, and creating a new class.

Problem 1: (5 points) Translate **this C++ program (https://camino.instructure.com/courses/87658/files/6535029?wrap=1) ↓ (https://camino.instructure.com/courses/87658/files/6535029/download?download_frd=1)** to Java (Quicksort)

Please name the file you submit HW3P1.java

Problem 2: (10 points) In this problem, you will explore inheritance. Implement the classes Food, Fruit, Orange, Apple, Seafood, Shrimp, and Prawn as represented in the **class diagram here. (https://camino.instructure.com/courses/87658/files/6535031?wrap=1)**

That is, Fruit inherits from Food, Orange inherits from Fruit, etc. In addition to what is shown, you may add a getter method for the name field in Food.

The first area in a class's box is its name, the second is its member data fields and their types, and the third is its member methods. So you can tell what the methods should do, I've provided **a main function and its output. (https://camino.instructure.com/courses/87658/files/6535036?wrap=1) ↓ (https://camino.instructure.com/courses/87658/files/6535036/download?download_frd=1)** **Note that the name field of any object of a given type should be initialized to match its class name; ex: Apple objects' names should be initialized to "apple".**

Please submit a .java file for each class, and the class with your main function should be called HW3P2.java

Problem 3. (5 points) We started on a BinarySearch Class in class on Monday (Code is on Camino). In this problem you will make it "more" object-oriented in the following ways:

-Note its name is now SortedList

-You will make a constructor that takes no arguments initializes the member field

-You will add a constructor that takes in an initialized array

-You will add an insert function that adds a value to the list and maintains its sorted-ness.

-You will add a quicksort function (feel free to use what you wrote #1).

-You will add a print function that prints out the contents in a nice format.

-You will make updates as necessary to the main function so that it still runs and tests your code. Up to you if you want to separate it out into a different class.

Please name your file for this problem SortedList.java

Problem 4 (10 points). You'll write a function which takes as input an ArrayList of Strings and Integers, and returns an ArrayList of objects from Problem 2 (to make it simpler, we'll restrict it to Food, Fruit, Apple, and Orange objects), and a sensible main that tests it. The starting ArrayList should be formatted as follows: the name of the type of the next object, followed by parameter(s) that should be sent as input to the class' constructor (however if you should send a Date as input, you will be given either the string "now" if you should use Date's constructor that takes no parameters, or an Integer if you should use the constructor that takes a long). So for instance, the input ArrayList could contain the following: "Food", "Orange", "Fruit", "Papaya", Integer(3932728), "Apple", "now", "Food", "sandwich", and the resulting ArrayList would have 5 objects in it, of type Food, Orange, Fruit, Apple, and Food.

You may wish to refer to the documentation for the Date class: **https://docs.oracle.com/javase/7/docs/api/java/util/Date.html (https://docs.oracle.com/javase/7/docs/api/java/util/Date.html)**

For this problem, you do not need to re-submit your Food, Fruit, Apple, and Orange classes. Simply submit the file containing your function and a main that tests it. The file should be named HW3P4Lastname.java where Lastname is your last name.