CIS 2571: Introduction to Java
Lab Assignment

Name: _____

Lab Assignment	#9 – Abstract Classes and Interfaces
Due Date *(beginning of class)*	04/16/2014
Points	Short Answer Questions _____/ 30 pts. Interfaces Java Program source code • AccountWithInterface.java (*modified Account.java file*) • TestAccountWithInterface.java (*new*) output • account array before and after sort uploaded .zip _____/ 40 pts. **Total _____/ 70 pts.**

Lab Assignment #9 Activities

1. Answer the questions in the spaces provided or attach a sheet with your answers (be sure to label questions for proper credit) *(2 points each, unless noted otherwise)*:

Question	Answer
a. The **private** accessor type is used for an **abstract class** constructor method.	**True** or **False** *(circle one)*
b. An **abstract class** must have at least one **abstract method**.	**True** or **False** *(circle one)*
c. An **abstract class** can be inherited from a **concrete class**.	**True** or **False** *(circle one)*
d. **Abstract classes** can be used to create object reference variables.	**True** or **False** *(circle one)*
e. Instance variables of a **concrete** subclass can be assigned to reference variables of the **abstract** superclass type.	**True** or **False** *(circle one)*
f. **Interfaces** can have instance data fields.	**True** or **False** *(circle one)*
g. A class can implement more than one **interface**.	**True** or **False** *(circle one)*

h. When a class implements an **interface**, it must implement <u>all the methods</u> defined in the interface with the exact signature and return type.	**True** or **False** *(circle one)*
i. The **private** accessor type is used for an **interface** constructor method.	**True** or **False** *(circle one)*
j. What is the **method header** of the abstract method to be overridden when the **Comparable** interface is implemented? *(4 points)*	
k. What is the **method header** of the abstract method to be overridden when the **Cloneable** interface is implemented? *(4 points)*	
l. **Interfaces** can <u>extend</u> other **interfaces**.	**True** or **False** *(circle one)*
m. **Wrapper** classes are immutable.	**True** or **False** *(circle one)*

2. You will modify a version of the **Account** class from previous labs to incorporate interfaces and use a generic sorting algorithm. The **Account.java** file will be given to you to modify. The modified class should be called **AccountWithInterface**.

   a) The **AccountWithInterface** class should implement the following interfaces: *(20 points)*

   - **Cloneable → deep copy** should be done for the **dateCreated** field.
   - **Comparable → balance** field should be used for comparison.

   b) Create a **TestAccountWithInterface** class with a main method that accomplishes the following program requirements: *(14 points)*

      - Create an object called **accountTemplate** of the **AccountWithInterface** class type. Use your own data for the annualInterestRate, id, and name fields. Create an array of three (3) **AccountWithInterface** objects called **accountArray** that have been **cloned** from the **accountTemplate** object.

- Set the balances for the **accountArray** elements as follows:

  accountArray[0] → 85900.32
  accountArray[1] → 3250.99
  accountArray[2] → 6200.56

- Use a loop to print out all the elements of the array using the class **toString**() method.
- Sort the **accountArray** using any sort method that takes advantage of the implemented **compareTo()** method of the **AccountWithInterface** class.
- Use a loop to print out all the elements of the *sorted* array using the class **toString**() method.

c) Add a block comment at the top of each file to identify your name, file, date, class, assignment, and short description of the included class. **Use proper code alignment for full credit.** *(2 points total, 1 point each)*

d) Compile the source code until no errors are found.

- Common Errors: http://www.cs.armstrong.edu/liang/intro9e/debug.html

e) Run the Java bytecode and observe the results.

f) Attach a hardcopy printout of your source code files.

g) Attach a hardcopy printout of your sample output. *(2 points)*

h) Create a **.zip** file containing only your **.java** source code files. Upload a copy of this **.zip** file to the appropriate assignment in Blackboard (this will be demonstrated during class, if necessary). *(2 points)*

- See the following link for a video on submitting assignment in Blackboard:
  http://ondemand.blackboard.com/r91/movies/bb91_student_submit_assignment.htm
- See the following link for a video on creating **.zip** files in Windows XP:
  http://www.youtube.com/watch?v=3xqF56OZo_k
- See the following link for instructions on creating **.zip** files:
  http://condor.depaul.edu/slytinen/instructions/zip.html