**CSE 220 Homework Assignment 2 (Due 4/9/23)**

**1) (30 pts)** Your first task deals with constructing a class hierarchy diagram. Recall that in class we looked at a simple example of different types of Animals in an inheritance hierarchy. **Construct a class hierarchy diagram for a type of physical technology of your choice (ex: variants of personal computers, phone technology), given the following constraints: there should be at least six classes ii. there should be at least three inheritance relationships** **depicted**. **iii.** **No more than half your classes should be abstract (if any)** **Note: you are to construct a class hierarchy diagram -- not a UML Class diagram!** If you are unsure how to construct a class hierarchy diagram you should follow the example provided for Animals in the Chapter 9 Part 2 presentation. You do not need to specify variables or methods for the classes.

**2) (30 pts)** Your second task deals with inheritance and *overriding methods*.   
a. Explain the difference between method overloading and method overriding in object-oriented programming.   
b. Can a constructor for a class be overloaded? Explain your answer.   
c. Can a constructor for a class be overridden? Explain your answer.   
d. Explain what the keyword **final** is in Java, and how it relates to inheritance and overriding.

3) **(40 pts)** You are to write three hierarchical classes in Java with the specifications below.

a. The *abstract* **Polygon** class consists of the following:  
i. an ArrayList of *Integers* (remember that ArrayLists cannot store primitive data directly) named **sides**, which represent the edges/sides of the polygon: sides should be accessible to inheriting classes, but not public.  
ii. a constructor that takes in an ArrayList of Integers as a parameter and uses it to initialize sides  
iii. a method **int computePerimeter()** that returns the polygon’s perimeter  
iv. an *abstract* method **double computeArea()**.

b. A **RightTriangle** inherits from Polygon and represents a right triangle with the first two sides in the ArrayList representing the legs, and the third representing the hypotenuse. It has the following additions/changes:  
i. A constructor that must verify that the input ArrayList has exactly three elements, and that the three satisfy the Pythagorean theorem (a2+b2=c2), where a and b are the first two sides, and c is the third. If either condition is failed, an exception must be thrown. Note: for convenience you can assume any right triangles we test your code with will use values that satisfy the Pythagorean theorem exactly (examples include 3, 4, and 5; 12, 5, and 13; etc.)  
ii. **double computeArea()**  should return the correct area of the right triangle.

c. A **Rectangle** inherits from Polygon and represents a quadrilateral whose first and third sides have equal length, and whose second and fourth sides have equal length. It has the following additions/changes:  
i. A constructor that must verify that the input ArrayList (sides) has four elements, and that side 1 and 3 match, as well as that side 2 and 4 match. If any of these three conditions is failed, an exception must be thrown.   
ii. **double computeArea()** must correctly return the area of the rectangle.

*Reminders regarding exceptions:  
i. A constructor that throws an uncaught exception must indicate this explicitly. For example, your signature for the RightTriangle constructor should likely be:****public RightTriangle(ArrayList<Integer> sides) throws Exception   
{  
<your code here>  
}****ii. You can throw the exception using syntax* ***throw new Exception(<message>);*** where <message> is any appropriate string indicating invalid parameter values were used (Ex: “A triangle must have exactly three sides.”)

**Responses to Problems #1 and #2 should be in .doc(x) or .pdf format. Upload a .zip file containing this file and your .java files for problems 3 to Blackboard as your assignment submission.**

**Your submitted file should have the filename “LN\_FN\_4.zip” where LN is your last name and FN is your first name.   
*Remember: do not submit your .bluej files or .class files: they have no source code!***