Test 1 – Version 2

Last, First Name		Student ID	

Question 1: Theory Questions [10 points total]

List all the engineering benefits of Commenting as documentation [5 points]

> Four things -- Provides reminders for what the code means/trying to do, reduces the learning curve of new developers, simplifies the writing of manuals, provides a way to trace back development milestones.

List all the engineering benefits of Correctness. [5 points]

> Three things -- Developer confidence in quality of code, user confidence in quality of information returned by the software, reduces systemic run-time errors (recurring hidden/unknown errors)

Question 2: Well-designed object 1 [20 points]

A programmer wants to create a tool that can stack any object and compare the equality of any two stacks. Each stack can contain any object type. This means that objects currently in the stack may not be from the same class, for example a stack may contain House, Car, Dog objects. The programmer does not want to use the class Object in the solution.

The class is called StackTool. It implements both the stack and the compare two stacks code. All stackable objects must contain the method String getKey() that returns a string containing information about the object. This information will be used to compare the equality between two objects. The compare method is static and takes two objects as parameters. When comparing the equality of two stacks the class contains a method that checks if every element in each stack is the same, however it is not knowable what class type the user will put in the stack. The user of StackTool will need to write the compare method that the compare stack method will call. This will use the getKey() method on each object, returning true when the objects are the same and false when they are not. The same, means, that the string returned by getKey() has a specific rule important to the developer/user expressed in the compare methods they provide.

Do not write this entire program. Do not use English (French) to explain how you would do it. Write code snippets. Be complete. Show off the techniques you used to make your solution well-designed. This is a Java programming question.

> SEE VERSION 1 FOR THE SOLUTION – exact same question

Question 3: Well-designed object 2

Do not write comments, but create a well-design object in every other way. This question asks you to implement a FIFO queue of dog sizes. The queue holds a maximum of 100 dogs. You are only writing the queue class. Assume there is an unsafe calling environment accessing your class. Valid dog sizes are integer from 0 to 10. The class provides the following public methods: enqueue (add a size to the end of queue), dequeue (remove and return a size from the front of queue), peek (do not remove size but return the size from the front of queue). You can write supporting methods. Write this class and all its members using all the (as needed) good design principles we saw in class. This is a Java programming question.

> SEE VERSION 1 FOR AN EXAMPLE OF HOW THIS SOLUTION SHOULD LOOK LIKE
>
> This does not need to be a circular array, but if the student implemented it that way, then even better.
>
> Notice the private portion of the class and the public portion of the class in the solution of Version 1. The student needs to do similar here.
>
> Notice how some testing is through return values while others are through exceptions. Version 1 is very similar to Version 2, so they should be implementing very similar methods.