CSSE 374 - Lab 1-1 Responses

1. *Identify and explain problems prevalent in the students’ design.*

* Student’s design favors inheritance over composition
* Changes to the DataStandardizer class can unintentionally affect the behavior of all DataStandardizer subclasses.
* If more subclasses are used to extend functionality in the future, then it will be difficult to gain knowledge of all DataStandardizer parsing behaviors.
* The parse behavior of a DataStandardizer cannot be changed at runtime.
* Subclasses of DataStandardizer that share a common (non-default) parse behavior will be forced to duplicate each other’s code.
* If more DataStandardizers are built by subclassing AmazonDataStandardizer and then support for Amazon data suddenly becomes unnecessary, major restructuring of the project must occur.
* The design does not encapsulate what is expected to change.
* The design does not separate what changes from what stays the same.
* If another behavior besides parse() needs to be added and maintained, behavior will have to be maintained in each DataStandardizer subclass.
* Insisting that everyone use a new subclass may not be realistic. Other developers may forget to update their code because it works fine with the existing DataStandardizer.
* DataStandardizer simultaneously supports two different standards by default (*Google* and *Microsoft* – *if the form is unknown*). This violates the single responsibility principle.

1. *Explain why sub-classing DataStandardizer and overriding the parse() method is not the best idea.*

What if someone else needs to add support for yet another format? Which class does that person subclass? What if those people don’t talk to each other?

1. *Create a UML Class Diagram to present your new design idea and explain it in a few lines.*

**

*Add parseFirstLine() to DataStandardizer and use that to switch between ParseBehaviors.*