Name: ________________________________ Date:____________

Midterm (Part 4)

Objectives

Part 1 – SOLID Principles (50 Points)

Part 2 – Questions (30 Points)

Part 3 – UML Diagram (20 Points)

Part 4 – Refactoring (100 Points)

Total – 200 Points

Background/Scenario

Complete the midterm in a Microsoft Word document.

Midterm is open notes, open book, open computer, and open internet. However, it is not open other students. Please refrain from discussing or collaborating with other students on the midterm questions.

While this is not a writing class, I will not mark you down for spelling or grammatical errors, but I do expect you to at least reread your answers and check for basic mistakes. However, if the grammar or spelling is so bad that I cannot understand what you wrote, I will have to mark it incorrect.

The midterm is worth 20% of your grade.

Required Resources

- Microsoft Word
- Visual Studio 2017
- Midterm Starter Code Solution

Note:

Parts 1 – 3 are omitted from this document.

Part 4: Refactoring Activity (100 Points)

For this part, you will need to download the midterm starter code package from Canvas. Once you have downloaded and extracted the package, save the solution to your Labs repository and immediately commit and push your commit.

While the starter code builds and runs, the state of the code isn't that good--it is your task to refactor it so you can add functionality. Because refactoring always has the potential to break functionality, you will also write unit tests to verify that what you refactored still works.

Refactoring Activity Rubric

Item	Points
Correct Output • Program output matches expected • No compiler errors or warnings	15 Points
Unit Tests • Unit tests for all business rules • At least two cases for each	25 Points
Refactoring • Code refactored to allow for unit testing • At least two design patterns have been implemented • At least two instances where SOLID principles have been implemented • Incremental Git commits for every milestone change you make with an appropriate commit description explaining your changes.	40 Points
Reflection • In the same Word document with Parts 1 – 3 answers of your midterm responses, include: • a single paragraph each explaining the two design patterns you have chosen to implement and where you implemented it • a single paragraph each explaining the two SOLID principles you followed and where you implemented it.	10 Points
UML Diagram • Create a UML class diagram of all of the classes and their relationships once the refactoring is completed and new functionality has been implemented.	10 Points
Total	100 Points

Important! All code changes must be commited incrementally to your Labs Git repository! Just like all the math classes you have taken in the past, your commit log is showing proof of work. If you only have a final finished product commit with no work to show for it, you will receive a zero.

Gilded Rose

Introduction

Hi, and welcome to team Gilded Rose. As you know, we are a small inn with a prime location in a prominent city ran by a friendly innkeeper named Allison. We also buy and sell only the finest goods. Unfortunately, our goods are constantly degrading in quality as they approach their sell by date. We have a system in place that updates our inventory for us. It was developed by a no-nonsense type named Leeroy, who has moved on to new adventures. Your task is to add a new feature to our system so that we can begin selling a new category of items. First an introduction to our system:

- All items have a SellIn value which denotes the number of days we have to sell the item
- All items have a Quality value which denotes how valuable the item is
- At the end of each day our system lowers both values for every item

Pretty simple, right? Well, this is where it gets interesting:

- Once the sell by date has passed, Quality degrades twice as fast
- The Quality of an item is never negative
- "Aged Brie" actually increases in Quality the older it gets
- The Quality of an item is never more than 50
- "Sulfuras", being a legendary item, never has to be sold or decreases in Quality
- "Backstage passes", like aged brie, increases in Quality as its SellIn value approaches; Quality increases by 2 when there are 10 days or less and by 3 when there are 5 days or less but Quality drops to 0 after the concert

Just for clarification, an item can never have its Quality increase above 50; however, "Sulfuras" is a legendary item and as such its Quality is 80 and it never alters.

We have recently signed a supplier of conjured items. This requires an update to our system:

- "Conjured" items degrade in Quality twice as fast as normal items
- “Enchanted” items degrade in Quality half as fast as normal items.

Feel free to make any changes to the UpdateQuality method and add any new code as long as everything still works correctly. However, do not alter anything inside the GoblinLib or the Inventory property in the Program.cs as those belong to the goblin in the corner who will insta-rage and one-shot you as he doesn't believe in shared code ownership (you can make the UpdateQuality method and Inventory property static if you like, we'll cover for you).

Part 5: Midterm Score

Part	Points Earned	Points Possible
Part 1		50
Part 2		30
Part 3		20
Part 4		100
Total		200

Grade: