TP5: Good practices, Refactoring, Testing, API Call

Bertrand Maxime

Part 1:

1) What is this code about?

This code is an inventory which changes every day by modifying the quantity of products in the inventory. All items have a SellIn value which denotes the number of days we must 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. The daily updates depend on the products. Once the sell by date has passed, quality degrades twice as fast. The quality of an item is never negative. "Aged Brie" 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 it's 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.

2) Can you clearly identify the name of all the goods stored in the Guilded Rose?

The names of all the goods stored in the Guilded Rose are:

- Aged Brie
- Sulfuras
- Backstage passes
- Elixir of the Mongoose
- +5 Dexterity Vest
- Conjured Mana cake

3) What happens when the day is over?

When the day is over, the code deals with the quality of the products and does -1 or +1 in function of the characteristics described before.

4) What happens to cheese when the day is over?

When the day is over, the cheese has more quality than the day before even if his sellin is negative cause this cheese increases in quality the older it gets.

5) What happens when a concert ticket goes over its expiration day?

If the concert ticket goes over its expiration day, his quality goes immediately at the value 0.

6) What makes this code hard to read?

This code is hard to read because there are too many lines of code while at least half of it could be reduced and rearranged. Also, there's no commentaries to better understand what the person who coded wanted to do. Moreover, the variables are very different and lack of precision.

7) Do you think the rules are clear enough so that you could rework the entire solution from scratch?

I think there are some information that are missing in the rules. Most of them are present so it could be possible to recreate it but it will be long and some things will miss, especially that the information in the rules are not well explained.

Part 2:

In part 2 we must create tests to verify the limits of the program. To test a program is always important to find out the errors in the code and know where it comes from exactly. Once the error is found, it's easier to correct it, we win time and complexity.

Part 3:

Now we can change the code (wasn't allowed before part 3). So, the of else statement in UpdateQuality() function is the function I decided to modify/add corrections. Adding a switch case statement seems to be a clever way looking at the main problem we have, readability.

Part 4:

The conjured item part. Here we deal with the conjured version of the items. It's still in the switch case statement and we test if the item is conjured or not.