Voyager Technical Test Documentation

Cameron Smith

Project Overview

Project

Voyager Technical Challenge: Point of Sale system with Testing

Description

Implement a class library for a point-of-sale scanning system that accepts an arbitrary ordering of products, similar to what would happen at an actual checkout line, then returns the correct total price for an entire shopping cart based on per-unit or volume prices as applicable.

Overview

Showcase the VoyagerPOS Terminal and unit tests. Well structured OO code written in C#

Technologies/Collections Used

VoyagerPOSTerminal

Framework - .NetCore 3.1
ClassLibrary
IDictionary - storting data which are linked by a key and value, string being the key and the total calculated price.
IList - Displaying a set of strings, which are ordered.
.Union - Compares values (Price)
.Orderby - Lists in a descending order by a key (PriceForUnit)
Ienumerable - In other words, something is countable if it *has a counter*. And the counter must basically: remember its place (*state*), be able to *move next*, and know about the *current* person he is dealing with. Enumerable is just a fancy word for "countable". In other words, an enumerable allows you to 'enumerate' (i.e. count).

VoyagerPOSTests

Unit Tests:

```
[TestCase("", ExpectedResult = 0.00)]
[TestCase("B", ExpectedResult = 4.25)]
[TestCase("BB", ExpectedResult = 8.50)]
[TestCase("ABCDABA", ExpectedResult = 13.25)]
[TestCase("CCCCCCC", ExpectedResult = 6.00)]
[TestCase("ABCD", ExpectedResult = 7.25)]
```

Dependencies

.NetCore
Nunit
NUnit3TestAdapter
Microsoft.NET.Test.Sdk

Variables/Classes/Methods

List of variables used in VoyagerPOS.

Variables

- code
- totals
- volume
- applyDiscount
- adjustedFinalPrice
- volumeAmount
- newVolume

Classes

- PointOfSaleTerminal
- ItemTotalCalculator
- Accumulator Record
- Total (Total.cs)
- VolumePrice (VolumePrice.cs)
- Tests (VoyagerPOSTests)

Methods

- SetPricing
- Scan
- ApplyBulkDiscountIfNeeded
- SetTotalDiscount
- CalculateDiscount
- SetPricing
- SetUp
- Sacnning_Product_Code
- CalculateTotal
- ApplyingBulkDiscount

What I learnt

Before doing the technical test at voyager this is what I had to learn to complete the challenge:
IEnumerable
IDictionary
Ilist
.Unions
.OrderBy
C# Unit Testing

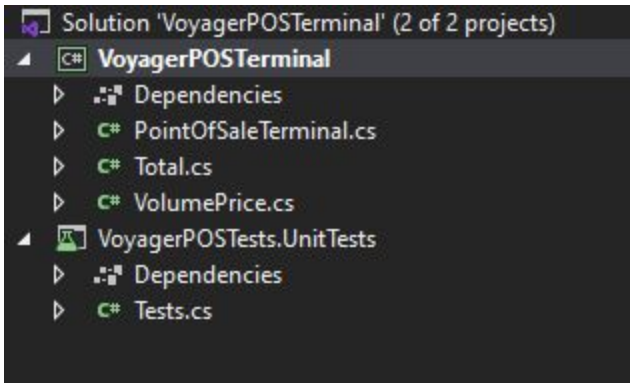
Was a great learning experience and I really enjoyed the technical test. It also showed me that what I was taught was really needed to overcome it.

Problems Faced

To learn many new techniques and methods within C# and the .Net Framework, resources used during this challenge were found through Google, stack over flow and GitHub. In the Test explorer having <unknown project> showing instead of VoyagerPOSTests.UnitTests

System Architecture

Class Library
PointOfSaleTerminal.cs
Total.cs
VolumePrice.cs
Unit Test
Tests.cs



Data Dictionary

Product Code Unit Price Bulk Price
A $1.25 3 for $3.00
B $4.25
C $1.00 $5 for a six-pack
D $0.75

Use Cases

Consider a grocery market where items have prices per unit but also volume prices. For example, Apples may be $1.25 each, or 3 for $3.

ABCDABA Product codes are scanned with a total being = $13.25 user runs a unit test with the results being [TestCase("ABCDABA", ExpectedResult = 13.25)]

CCCCCCC Product code is entered with a bulk discount being applied with the total being = $6
[TestCase("CCCCCCC", ExpectedResult = 6.00)]

ABCD Product codes are scanned with a total price of $7.25
[TestCase("ABCD", ExpectedResult = 7.25)]

Test gives back the results to the user with all seven passing.

Software Design Patterns

Factory Method - **Identification:** Factory methods can be recognized by creation methods, which create objects from concrete classes, but return them as objects of abstract type or interface.

Advanced POS System

I also attempted a more advanced POS system following yarchi's POS system on github. Ran into issues with the calculation, but was still a great learning experience.

References

https://docs.microsoft.com/en-us/dotnet/core/testing/unit-testing-with-nunit
https://nunit.org/documentation/
https://github.com/yarchiT/POS-Terminal
https://refactoring.guru/design-patterns/factory-method/csharp/example#example-0
https://stackoverflow.com/questions/558304/can-anyone-explain-ienumerable-and-ienumerator-to-me