Soda Machine User Stories

Out of 85 points

Using the concepts of OOP by creating classes and using objects (instances of those classes) to interact with each other, create a console application representing a Soda Machine that takes in payment and dispenses sodas.

User stories:

(5 points): As a developer, I want to make good, consistent commits.

(5 points): As a developer, I want to account for and handle bad user input, ensuring that any user input is validated and reobtained if necessary.

IT IS RECOMMENDED FOR THE BELOW USER STORY TO BE COMPLETED FIRST (AFTER CREATING CLASSES FROM PROVIDED UML DIAGRAM)
(25 points): As a Customer, I want the following results to occur when attempting to purchase a soda:

- If enough money is not passed in, don't complete transaction and give the money back.
- If exact change is passed in, accept payment and dispense a soda instance that gets saved in my Backpack.
- If too much money is passed in, accept the payment, return change as a list of coins from internal, limited register, and dispense a soda instance that gets saved to my Backpack.
- If too much money is passed in but there isn't sufficient change in the machine's internal register, don't complete transaction: give the money back.
- If exact or too much money is passed in but there isn't sufficient inventory for that soda, don't complete the transaction: give the money back.

(5 points): As a developer, I want my soda machine to start with the following inventory:

- Coins: 20 quarters, 10 dimes, 20 nickels, 50 pennies
- Cans (you pick how many of each the machine starts with): Root Beer (60 cents per can), Cola (35 cents per can), and Orange (6 cents per can)

(5 points): As a developer, I want my Coin classes to have a read-only property for double value (public property & protected field for member variable double value).

(10 points): As a developer, I want a static user interface class to allow the user to be prompted to type in selections. All Console interactions (ReadLines and WriteLines) should be in this class.

(10 points): As a Customer, I want to keep track of my Coins in a Wallet class and my Cans in a Backpack class. Backpack should start empty; Wallet should start with at least $5 in mixed change.

(10 points): As a Customer, I want to select the coins I'm entering as payment and have them added to a List.

(5 points): As a Customer, I want to choose which soda to be dispensed from the current inventory of the machine.

(5 points): As a developer, I want to use C# best practices, SOLID design principles, and good naming conventions on the project. This includes proper usage of Public/Private variables and methods.

BONUS:

(5 points): As a Customer, I want to use Method Overloads to allow for payment with a Credit Card in addition to coins.