**Out of 90 points**

**User stories:**

As a developer, if I don’t know what Lemonade Stand game is, I will play the game online for a bit to get familiar with the gameplay.

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

**(20 points): As a player, I want the basic Lemonade Stand gameplay to be present.**

**(10 points): As a player, I want a weather system that tells me what the weather condition and temperature is at the beginning of each day.**

**(10 points): As a player, the price of product as well as weather/temperature should affect demand, so that if the price is too high, sales will decrease, or if the price is too low, sales will increase, etc.**

**(10 points): As a player, I want each customer to be a separate object with its own chance of buying a glass of lemonade, so that how much lemonade is purchased and how much a customer is willing to pay will vary from customer to customer.**

**(5 points): As a player, I want the ability to make a recipe for my lemonade, so that I can include x-amount of lemons, x-amount of sugar, and x-amount of ice.**

**(10 points): As a player, I want my game to be playable for at least seven days.**

**(10 points)**: As a developer, I want to implement the SOLID design principles as well as C# best practices in my project, so that the project is as well-designed as possible.

Need a better default case in Game class SellLemonade method.

**(10 points (5 points each))**: As a developer, I want to pinpoint at least two places where I used one of the SOLID design principles and discuss my reasoning, so that I can properly understand good code design. Minimum of two SOLID design principles must be used.

Weather class – line 35

Below is an example of Open/Closed SOLID principle. Instead of hardcoding the number 5 in the rnd.Next(0, 5) I used weatherConditions.Count so that in the future I can change the number of weather conditions without having to change the code.

Player Class – line 56

This is an example of the Single Responsibility SOLID principle. I isolated the single act of getting a player’s name in its own function.

**Classes You Will Use (you may need more than what is provided):**

Program

Weather

Customer

Game

Inventory

Player

Store

Day

UserInterface