**Lemonade Stand Planning Phase**

|  |
| --- |
| In many professional situations, you will be working as part of a scrum team. As you plan for a sprint, your team will start by developing goals and describing the steps it will take to complete the project. For Lemonade Stand, you will explore this first phase in development; research and planning. |

Research and Explain Game Function

Find a version or two of lemonade stand to play online (some rules will vary between versions). Then, list the steps of the game. Make sure to include the step that initiates the game, and the step that will end the game. Refer to your User Stories as some of the framework for the game is already defined. (i.e. number of times your game will run before giving the final result).

* <https://www.coolmathgames.com/0-lemonade-stand>
* Introduction / Explain the game
  + 7/14/21 days (optional, per user stories); Make as much $$ as possible; have control pricing, Q/C, inventory, purchasing – start with $20
  + Recipe important
  + Watch weather / anticipating sales, adjusting purchasing
  + Customer satisfaction
* Choose length of game – 7/14/21 days, more??
* All screens after this has the forecast high temperature & forecast conditions – hazy, cloudy, rain, others?, money in the bank, & the day count
* Inventory / purchasing screen – count of cups, count of lemons, cups of sugar, ice cubes;
  + - consider adding salt (can advertise it has electrolytes = good for runners/people exercising, adding a bit of salt & more sugar brings out flavor (learned in a Chinese cooking class – add both sugar & salt to enhance flavor), also might make them come back for 2nd cup
    - consider adding option to purchase signs
  + Has buttons to purchase each of the 4 items; add button for salt,
  + Help screen for inventory & purchasing
  + There is a **bankrupt button**, probably for leaving the game early, has end of season & profit/loss reports, play again button, amount of money left if any
  + Has OK button, with which you progress to the Price / Quality Control screen
* Price/quality control (Q/C) screen, where you set the price & recipe
  + Price per cup, defaults to $0.25
  + Lemons per pitcher, defaults to 4
  + Cups of sugar per pitcher, defaults to 4
  + Ice per cup, defaults to 4 cubes
  + Has Back to Store button, to purchase more supplies
  + Has OK button, with which you progress to start the game, for the day
* Game screen has slider to set the price on the fly, shows people going by, some purchasing/drinking, elapsed time from 8?am, & remaining #cups #cubes #lemons #cups of sugar, & #cups in the pitcher remaining
* End of day report shows
  + how many cups you sold to how many potential customers
  + considering the weather, I’d say this is great/?average/?poor
  + customer satisfaction percentage
  + popularity percentage (which grows as your customer satisfaction stays high??)
  + okay button to go to inventory losses screen
* Inventory losses screen –
  + Always shows remaining ice has melted,
  + Possible lemons that have spoiled
  + Possible bugs in the sugar
  + ? suspect there is a wind storm that would blow away your paper cups, signs you may have purchased
  + Has okay button, to start another day, goes to inventory/purchasing screen

Steps to the game:

1. Initiate the game
2. Display/read the rules, advice
3. Click okay to continue
4. Enter # of days to play / list option,
5. Click okay to continue
6. Display User Decision Screen –
   1. Day Number
   2. Day’s forecast temp & conditions
   3. current temp & conditions
   4. recipe
   5. price per cup
   6. inventory/purchase ingredients with quantity/amount available at the store
   7. number of pitchers producible (is that a word?) based on current recipe & inventory
   8. money on hand
   9. number of pitchers you want to (be able to) make
   10. quit option
7. User can change any of the player-choice settings (change recipe, purchase ingredients)
   1. Any change/purchase clears the screen and redisplays the User Decision Screen
8. Enter special key to continue to play a day’s round based on current settings
9. Game/algorithm determines how many potential customers become actual customers, based on:
   1. Price – what’s optimal for weather conditions?
   2. quality of lemonade – what’s optimal recipe?
   3. Mood – based on weather conditions
10. Display Daily Report Screen – very similar to User Decision Screen
    1. today’s forecast & actual high temp & conditions
    2. number of cup sales
    3. current sale price per cup
    4. cost per pitcher
    5. how many pitchers & total cups of lemonade thrown away = waste
    6. today’s forecast high temp & conditions
    7. profit or loss for the day’s activity
    8. Money on hand
    9. press enter/return to continue
11. Repeat steps #6 through 10 for each day
12. Display End of Game screen
    1. Totals for all ingredients used, purchased, wasted(?)
    2. Total sales
    3. Final profit/loss
    4. Congratulations or sympathy message

NOTE: Consider upgrading & degrading the location of the lemonade stand for increased customer potential; i.e. start out in front of your house in the middle of your street; a friendly neighbor tells you to move in front of his house on the corner; a business owner tells you to move in front of his business; after a few days the city administrator tells you you don’t have a permit (because a friendly cop would just look the other way as he should) and shuts you down so you move back to the corner, etc.

See user stories, pasted at bottom

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Structure Classes

List classes you think you will need to include in your game. These may change as you develop your game.

Program – main entry, has

* + - game object
    - user interface object?

Game – has:

* + - runGame method
    - players
    - # of rounds/days
    - Weather object;

Player – has:

* + - choices of quantities of items to purchase
    - money
    - days/rounds
    - recipe for each day

Vendor/store

* + - Has inventories of each recipe ingredient
    - Each recipe item has sale price

Recipe

* + - Has ingredients
    - Has quantity of each ingredient
    - Has quantity of lemonade made(?) i.e. pitcher, gallon, ~ 12 x 10 oz serving

Pitcher – has:

* + - Capacity # of cups
    - Cups remaining?
    - Sell cup method
    - Fill method

Ingredient – has: (poss. good use of inheritance, all have name & price, not all have spoil/expiration i.e. cups)

* + - Name
    - Sale price per unit
    - canSpoil value (?) – perhaps for future use, future levels of difficulty
    - willSpoil value (e.g. ice)
    - Expiration Days(?)

Rounds/days object? – has:

* + - weather object
    - # potential customers
    - # actual customers / sales

Weather object; has:

* + - forecast & actual temperature
    - % chance of rain,
    - forecast & actual conditions
    - get forecast method(s) temp & conditions
    - get actual method(s) temp & conditions

Customer – has:

* + - isActual
    - mood
    - # of cups purchased

User Interface – has:

* + - Screens defined / laid out?
    - Input for various choices
    - Validation of input

Develop UML

Create a UML diagram that shows the connection between your classes as well as their properties.

Consider the following…

* What is the hierarchy between your classes?
* Where will you use inheritance?
* Where will the phases of the game be executed?
* Where/how will you handle user input?
* What member variables and methods will belong to each class?
* Which member variables and methods will be public or private?

There are many UML tools and you can use whatever is most comfortable. If you handwrite, give to instructor, if you complete in VS or another online site, send the document or a readable screen shot.

|  |
| --- |
| Save file as: LastName\_LSPlanning then Slack this document and UML to all instructors. You may start on your lemonade stand design after you have submitted. You will receive feedback on your plan as you begin working. |

========================= USER STORIES =========================================

**105 points total**

**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.

**(25 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 includes a forecast and actual weather, so that I can get a predicted forecast for a day or week and then what the actual weather is on the given 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 player, I want my daily profit or loss displayed at the end of each day, so that I know how much money my lemonade stand has made. I also want my total profit or loss to be a running total that is displayed at the end of each day, so that I know how much money my lemonade stand has made.

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

**(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.

**Bonus Points:**

**(5 points)**: As a player, I want the game to be playable for more than one player, so that I can have multiple humans play each other or a human play a computer.

**(5 points)** As a developer, I want to integrate a Weather API, so that my game has real-time weather based on a current temperature and forecast.

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

Program

Weather

Customer

Game

Inventory

Player

Store

Day

UserInterface