# Assignment05 – Lemonade Leftovers

In this assignment we will revisit the lemonade stand scenario we encountered in a previous activity.

This assignment has 2 parts.

In part 1, you will write code that will prompt the user to specify the amount of lemon juice, sugar, and cups are on hand.

Your calculator will determine how many cups of lemonade can be made, and how much of each ingredient will be left over.

In part 2, you will simulate the weather (and the corresponding demand for lemonade) over a certain number of days.

## Scenario 1

As seen in our prior activity, our recipe for one serving of lemonade is shown below.

### Lemonade recipe

For 1 serving of 240 ml (that is, about 1 cup)

* 190 ml water
* 50 ml lemon juice concentrate
* 80 grams sugar (about 3 ounces)
* 1 disposable cup

For this assignment, we’ll assume an infinite amount of water. But, if we were really budgeting for a lemonade stand, we would need to think about that too.

The calculator will always respond with the whole number of servings that are possible, given the specified quantities of concentrate, sugar, and cups. We are not interested in preparing half a serving.

But we do want to know many ingredients go unused. Your calculator will also determine the amount of leftover cups, concentrate, and sugar.

## Instructions (Part 1)

1. Retrieve assignment05.zip provided by your instructor.
2. Extract the code into your mis2402workspace and open the lemoncalc.html file in Visual Studio Code.
3. Watch this animation. This represents what you are expected to build in part 1.

<https://misdemo.temple.edu/classexamples/lemonade-calc.gif>
4. Notice that some input values are not accepted. Logically, you can’t have half a paper cup. For the sake of simplicity, you should assume that all your inputs must be natural numbers.
5. You should note that the number of servings you can prepare will be limited by the amount of sugar you have, or the amount of lemon juice you have, or the number of cups you have, *whichever is the most limiting*.

You need to stick to the recipe. You don’t’ have the option to make the lemonade more or less sweet!

1. You should also note that there are multiple div tags used to hold your output: textDisplayed1, textDisplayed2, textDisplayed3 and textDisplayed4.
2. A lot of coding has been done for you in the event handler. You should read the code in the event handler, ***but do not change it***. Instead, focus your efforts on these functions.

|  |  |
| --- | --- |
| **Function name** | **Comments** |
| isNaturalNumber | You may want to reuse earlier definitions of this function. It returns true if x is a natural number, and false if not. |
| calculateCups | Given the amounts of concentrate, sugar, and paper cups, determine how many cups (servings) can be prepared.If any of the inputs are not natural numbers, return “Bad data. Try again.” |
| leftoverconcentrate | Given an amount of concentrate, and a number of servings, return how much concentrate would be left over.If the value provided for the number of servings is not numeric, return 0. |
| leftoversugar | Given an amount of sugar, and a number of servings, return how much concentrate would be left over.If the value provided for the number of servings is not numeric, return 0. |
| leftovercups | Given a quantity of paper cups, and a number of servings, return how many cups would be left over.If the value provided for the number of servings is not numeric, return 0. |

**Continued**

## Scenario 2

Your solution will accept a certain number of days as input. Your code will then simulate the weather for that number of days, and the corresponding demand for lemonade. Here are some guidelines for our simulation.

* On an average day, the weather is “good”, and the demand for lemonade from your stand will be 100 cups of lemonade.
* On a very bad day there will be a no demand for lemonade at all. That is, the weather is extremely bad that day because of a thunderstorm, hurricane, flood, etc.
* On a bad weather day, you have demand for 30 cups less than average.
* On a good day, you have demand for 30 cups more than average.
* On a very good day, you have demand for 50 cups more than average.

Your solution should simulate the role of ***two six-sided dice*** to determine the weather. Use the following table to determine the weather for a day.

|  |  |
| --- | --- |
| **Value** | **Meaning** |
| 2 | Very bad weather. Zero demand. |
| 3-4 | Bad weather. Demand for 70 cups. |
| 5-9 | Average weather. Demand for 100 cups. |
| 10-11 | Good weather. Demand for 130 cups. |
| 12 | Very good weather. Demand for 150 cups. |

## Instructions (Part 2)

1. Edit the file lemondemand.html in Visual Studio Code.
2. Watch this animation. This represents what you are expected to build in part 2.
<https://misdemo.temple.edu/classexamples/lemonade-demand.gif>
3. Edit the functions found in the start file: isNaturalNumber, getDemand, and simulateDemand as described in the following table.

|  |  |
| --- | --- |
| **Function name** | **Comments** |
| getRandomInt | This function is given to you. Do not edit it. |
| isNaturalNumber | (Same as before.) You may want to reuse earlier definitions of this function. It returns true if x is a natural number, and false if not. |
| getAvg | This function takes two numbers, a total, and a value of n. It calculates the total, divided by n, rounded to two decimal places.No loop is needed in this function.No error trapping is needed in this function. |
| getDemand | Calculate a random number between 2 and 12 by simulating the roll of 2 six-sided dice. ***For the purposes of grading, you must use console.log in this function. Use console.log to display the value of the dice roll, and the corresponding demand.***This function should then **return an integer value** based on the roll. Possible outputs of this function are: 0, 70, 100, 130, 150. |
| simulateDemand | If the value provided for the number of days is not natural, return “Bad data. Try again.”If the number is good, loop for that number of days, calling the getDemand function to simulate the weather each day.This function will then return the total demand for the number of days specified. |

1. Test your work. Did you check the console log in the Chrome Web Developer tools? Do you see the console log statements being generated by the getDemand function. (Your instructor will be looking for those.)
2. Upload your work (both html files) to assignment 5 on canvas.

How will this assignment be graded?

Scenario 1 work is worth 50 points. Each function in part 1 needs to work as expected and all the functions need to work together to provide the desired result.

Scenario 2 work is worth 50 points. Each function in part 2 needs to work as expected and all the functions need to work together to provide the desired result.

Point deductions will be assigned in 5-point and 10-point increments.

Things you could lose credit for

* An individual function that does not work as expected.
* Forgetting to round output where directed.
* Forgetting to specify console.log output where explicitly requested by the instructor.
* Functions that do not work together to create the expected output.