DataBased Take Home Problems

Software Engineer Feb 2020

1 Instructions

1.1 Overview

There are three problems to solve. Please use Javascript or Python. Each problem should have it's own file. For example, if you are using Typescript, you would have 3 distinct .ts files (Not including test classes). Please include documentation on how to run each of your files, preferably in a README.md. You are allowed to use the internet and other resources to solve these problems. An acceptable use case example is looking up syntax. However, please do not plagiarize solutions. It is quite obvious when we review the solutions in person and it makes an awkward situation for both parties.

1.2 How do I know I am done?

There are no test cases provided, it is up to you to determine if your solution solves the problem. If you leverage TDD or test cases at all, please include your test files in your submission. We highly recommend including test classes in your submission.

1.3 Submission

We recommend having at least the following files included in your submission:

1. README.md
2. 1 file per problem
3. Test files (if applicable)

2 Problem 1 - Least Factorial

2.1 Problem Description

```
    function leastFactorial(n) {...}
```

Given an integer n, find the minimal k such that

k = m! (where m! = 1 * 2 * ... * m) for some integer m; k $\geq$ n. In other words, find the smallest factorial which is not less than n.

2.2 Examples

1. For n = 17, the output should be leastFactorial(n) = 24.
2. For n = 5, the output should be leastFactorial(n) = 6.
3. For n = 106, the output should be leastFactorial(n) = 120.

2.3 Constraints

$1 \leq n \leq 120$

3 Problem 2 - Recycling Lipstick

3.1 Problem Description

```
function getTotalNumberOfLipsticks(numberOfLipsticks, numberOfLeftoversNeeded){...}
```

You own a lipstick business. When a lipstick container is empty, there is actually some leftover lipstick at the bottom that cannot be used because it is not accessible. Being an environmentally friendly business owner, you would like to recycle the leftover lipstick to make more. As a business, you know you need 'numberOfLeftoversNeeded' to make a new lipstick. You have 'numberOfLipsticks' in your possession. What's the total number of lipsticks you can sell assuming that each of your customers return their leftovers?

3.2 Example

For numberOfLipsticks = 5 and numberOfLeftoversNeeded = 2 the output should be getTotalNumberOfLipsticks(numberOfLipsticks, numberOfLeftoversNeeded) = 9

Here is how you get 9 lipsticks: Sell 5 lipsticks, get 5 leftovers; Create 2 more lipsticks using 4 leftovers (1 leftover remains); Sell 2 lipsticks, end up with 3 leftovers; Create 1 lipstick using 2 leftovers (1 leftover remains); Sell 1 lipstick, end up with 2 leftovers; Create 1 lipstick using 2 leftovers (no leftovers remain); Sell 1 lipstick.

Thus you sell 5 + 2 + 1 + 1 = 9 lipsticks!

1. numberOfLipsticks = 5 numberOfLeftoversNeeded = 2 expected output = 9
2. numberOfLipsticks = 15 numberOfLeftoversNeeded = 5 expected output = 18

3. numberOfLipsticks = 2 numberOfLeftoversNeeded = 3 expected output = 2

4 Problem 3 - Students and Treats

4.1 Problem Description

```
function getLastStudent(numberOfStudents, treats, startingChair) {...}
```

A school teacher wants to hand out treats to his students. The teacher decides the best way to divide the treats is to have the students sit in a circle of sequentially numbered chairs. A chair number will be drawn from a hat. Beginning with the student in drawn chair, one treat will be handed to each student sequentially going around the circle until all treats have been distributed.

The teacher wants to have the students involved in sharing treats. He decides that whoever gets the very last treat, will be the student who makes the treats for the next game. Determine the chair number occupied by the student who will receive the last treat.

For example, there are 4 students and 6 treats. The students arrange themselves in seats numbered 1 to 4. Let's suppose 2 is drawn from the hat. Students receive treats at positions 2,3,4,1,2,3. The student who gets the last treat is in chair number 3.

4.2 Function Parameters

1. numberOfStudents: an integer, the number of students
2. treats: an integer, the number of treats
3. startingChair: an integer, the chair number to begin passing out treats from

4.3 Constraints

Notice the constraints for this problem. These are very large numbers, although it is highly unlikely there exists a class with that many students and treats, we still will be testing with large inputs.

1. $1 \leq$ n $\leq$ 10**9
2. $1 \leq$ m $\leq$ 10**9
3. $1 \leq$ s $\leq$ n

4.4 Examples

1. getLastStudent(5,2,1) = 2
2. getLastStudent(5,2,2) = 3
3. getLastStudent(7,19,2) = 6
4. getLastStudent(3,7,3) = 3

5 Problem 4 - Cinnabon Line

5.1 Problem Description

```
function  peopleWatch(heightOfPeople){...}
```

You are people watching at the mall and you notice that the Cinnabon line has people of various heights. Write a program that can accept a list of numbers for the height of each person in the line and returns the index for the next person in line that is taller than the person at that position. If there is no person taller than them in the line, return null (or in python None).

5.2 Example 1

Input: [5.5, 4.5, 4, 6, 3.3]
Output: [3, 3, 3, null, null]
Rationale: 6 is the next tallest in line for the first 3 people in line, after that no one has anyone behind them that is taller than them.

5.3 Example 2

Input: [6, 4.5, 5.5, 4, 6, 3.3]
Output:[null, 2, 4, 4, null, null]
Rationale: The first 6 has no one taller than them after them.

5.4 Constraints

1. Each person's height will be ≥ 1