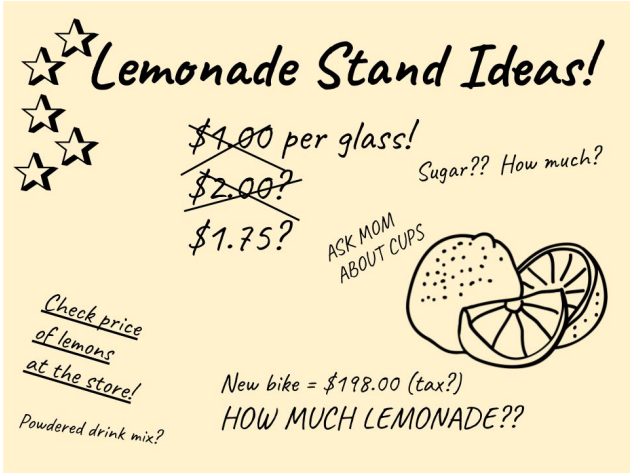
Problem Decomposition

@BootstrapWorld

Problem Decomposition

Sally runs a lemonade stand, which charges $1.75/glass. It costs her $0.30/glass to buy sugar, ice and lemons.

- What do you Notice?
- What do you Wonder?



Problem Decomposition

- Define functions for `revenue` and `cost` on Word Problems: revenue, cost. **Note: The information you need to write the `cost` function is provided in the Design Recipe word problem!**
- Once you've defined the functions, open Sally's Lemonade Starter File.
- Enter your code for `revenue` (including all examples and definitions) below the first prompt. Enter your code for `cost` below the second prompt. Click "Run" and make sure your tests pass.

Problem Decomposition

- What is the difference between *revenue* and *profit*?
- How could Sally *increase* her profits?
- What is the *relationship* between profit, cost, and revenue?

Problem Decomposition

- Complete Word Problem: profit, using the Design Recipe. (There are multiple correct solutions!)
- Add your code for `profit` in Sally's Lemonade Starter File below the third prompt; be sure to type all examples and definitions. Click "Run". Do all your tests pass?
- Optional: complete Sally's Bike

Problem Decomposition

This activity started with a situation, and you modeled that situation with functions. One part of the model was *profit*, which can be written several ways.

Turn to Profit - More than one Way! and reflect on the four function definitions presented.

Problem Decomposition

```
fun profit(g): (1.75 * g) - (0.3 * g) end

fun profit(g): (1.75 - 0.3) * g end

fun profit(g): 1.45 * g end

fun profit(g): revenue(g) - cost(g) end
```

Which of these four `profit` definitions do you think is "best", and why?

Problem Decomposition

Suppose the cost of lemons goes up. Which solution(s) would need to be changed?

What if Sally charges $2/glass? Which solution(s) would need to be changed?

```
fun profit(g): (1.75 * g) - (0.3 * g) end

fun profit(g): (1.75 - 0.3) * g end

fun profit(g): 1.45 * g end

fun profit(g): revenue(g) - cost(g) end
```

Problem Decomposition

`profit` can be *decomposed* into a simpler function that uses `cost` and `revenue`.

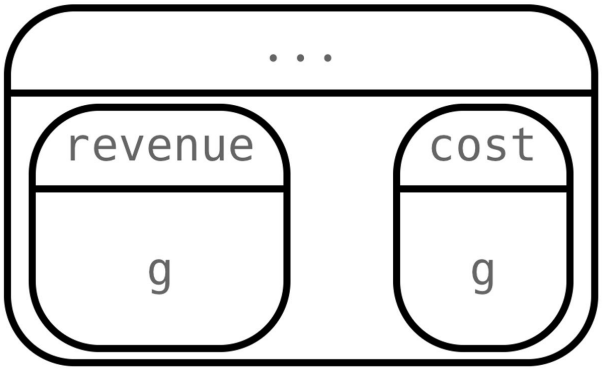
So what's the big deal?

1. Smaller pieces are *easier to think about*, and to test!
2. These pieces can also be *re-used*! Like lego pieces, smaller functions can be used to build all kinds of things.
3. Re-using code means *less code* overall. Less code means fewer places to make mistakes.
4. Re-using code means *less duplicate code*. When code needs to be changed, that change only needs to made in one place, instead of in multiple places.

Top-Down vs. Bottom-Up

Top-Down and Bottom-Up design are two different strategies for problem decomposition.

Bottom-Up: start with the small, easy relationships like `revenue` and `cost` first. How are they connected with the outer circle? You'll get there eventually, but **we can leave it blank for now (. . .)**. In the Lemonade Stand, you defined `cost` and `revenue` first, and then put them together in `profit`. *This is the same approach as building your Circle of Evaluation inside-out!*



Top-Down vs. Bottom-Up

Top-Down and Bottom-Up design are two different strategies for problem decomposition.

Top-Down: start with the "big picture" and then worry about the details later. We could have started with `profit` as `revenue - cost`, and **fill in the details of `revenue` and `cost` later (thus the `...`).** *This is the same approach as building your Circle of Evaluation outside-in!*

Top-Down vs. Bottom-Up

Jamal's trip requires him to drive 20mi to the airport, fly 2300mi, and then take a bus 6mi to his hotel. His average speed driving to the airport is 40mph, the average speed of an airplane is 575mph, and the average speed of his bus is 15mph. *Aside from time waiting for the plane or bus, how long is Jamal in transit?*

Take a moment to think: What would your first step be if you were trying to figure out how long Jamal would be transit? What circles would you draw or functions would you define to solve this? Would you work top-down or bottom-up?

Then turn to Top Down or Bottom Up.

Top-Down vs. Bottom-Up

- Whose strategy was Top-Down? How do you know?
- Do you have questions about either of these strategies?
- Which strategy to do you prefer? Why?