The Law of Large Numbers in Code

Bill the Statypus says: Manually tracking expected values on paper tells us the mathematical truth. But to see the Law of Large Numbers settle down in live production, we need a digital engine capable of processing thousands of trials instantly!

```
sample(x, size, replace, prob)
```

1. Code Grounding: Analyzing the Sister Script

Open RStudio and load your companion file: `SimulationsAndLLN.r`. Locate the code block under **TASK 1**. This script simulates an 8-hour shift consisting of $1,000$ customer coffee orders at *The Caffeine Statypus*, evaluating our syrup pumps random variable (X).

Reflection: Code Architecture Analysis

Execute the code block for Task 1 in your console. Evaluate the structural design parameters before making any manual updates:

1. Look at the script. Why must the `replace` argument be explicitly configured to `TRUE`? What would go wrong with coffee production after the 4th customer if it were changed to `FALSE`?

2. Look at the vectors. How does R's underlying logic automatically link an outcome element with its exact corresponding probability configuration weight?

3. Highlight and re-run Step 3 and Step 4 **together** three distinct times in your script window. Write down the three changing experimental means you receive:

Run 1: _____ Run 2: _____ Run 3: _____

Statypus Insight: Simulations are Random

The reason that we get different results is that `sample()` creates $1,000$ new customers each time based on the probabilities we passed to it. This is also why your values likely won't match your classmates' results.

2. Adaptation Challenge: The Espresso Line

Sally the Statypus says: Don't write new syntax from scratch! To evaluate a completely distinct operational variable, duplicate your known safe blueprint from Task 1, drop it into the Task 2 section, and adjust the values vector and trial size.

Recall your handwritten distribution table from the *ProbsAndProps* handout, where we mapped out a new random variable, Y, to track the number of **espresso shots** ordered per customer:

- Drip Coffee / Herbal Tea = 0 shots (Combined Probability = 0.50)
- Vanilla Latte / Caramel Macchiato = 2 shots (Combined Probability = 0.50)

Reflection: The Tweak Assignment

Copy your Task 1 blueprint block down into the **TASK 2** area of your R script file. Tweak only the input parameters to convert it into an espresso shot simulation.

1. Write down your newly modified vector definition for Step 1 below. How did you change the vector choices to represent random variable Y?

2. Did you need to alter your probability weights vector in Step 2 to reflect the values from your paper table? Explain why or why not based on the sample space outcomes:

3. Tweak your simulation's size configuration parameter up from **1,000** to **100,000** customers (`size = 100000`). Run your script. What experimental mean value does the console report?

4. Connect this behavior directly back to **Theorem 6.1 (The Law of Large Numbers)**. Why does scaling up the sample size force this experimental random average to stabilize onto the exact theoretical expected value ($\mu_Y = 1.0$) you found by hand?