INFO 190S

Project 1 Slayer Puzzle

Overview

One strategy for finding a solution to a number puzzle is to describe the solution by a set of algebraic constraints and then solve the constraints. Consider the following puzzle:

> *The Jets and the Sharks are the names of two rival dance teams. The Jets say to the Sharks, "If one of you joins our team, then our team will double the size of yours". The Sharks reply, "If one of you joins our team, then the sizes of our teams will be equal". What are the sizes of the two teams?*

If the sizes of the Jets and Sharks are denoted by *j* and *s*, respectively, then any solution to this puzzle satisfies the constraints:

- $j + 1 = 2(s - 1)$
- $s + 1 = j - 1$

Solving these constraints leads to a solution to the puzzle. For example, $j = 7$ and $s = 5$. While solving the constraints in this example is straightforward, solving constraints for some number problems can be hard. In such cases, another strategy is to guess possible solutions and then check if the guess works. Consider, the following puzzle:

> *What digits can replace the letters A, B, and C to make a 3-digit number ABC for which the following equation is true:* $ABC = A{\cdot}B{\cdot}C{\cdot}(A + B + C)$.

This puzzle is not easily solved directly. But any guess for the 3-digit number *ABC* can be easily *checked* to see if it satisfies the equation. For example, suppose you guess that *ABC* is *123*. This guess does not work because it requires $A = 1$, $B = 2$, $C = 3$, and $123 \neq 1{\cdot}2{\cdot}3{\cdot}(1 + 2 + 3)$. In contrast, the guess that *ABC* is *135* works since $135 = 1{\cdot}3{\cdot}5{\cdot}(1 + 3 + 5)$.

You will write a program that can be used to check guesses for the following puzzle:

> *For what 6-digit number SLAYER is the following equation true, where each letter stands for the digit in the position shown:* $SLAYER + SLAYER + SLAYER = LAYERS$.

Project Specification

You must write a program that does the following:

1. Output a brief description message (see examples below).
2. Prompt the user for a single guess for *SLAYER*. Your program should not ask for each digit individually.
3. Print out a message to the screen showing if the guess solved the puzzle.

Note: You must use only integers to solve this problem. You can't use strings or any string manipulation functions. You are to solve the problem using integers and integer operations only.

Getting Started

Open a text editor and create a new Python file named **p1.py**. You will write the entire solution for this problem in the **p1.py** file. You are only to submit **p1.py** to Gradescope.

Restrictions

Your solution must only use the material we have covered up to this point in class. For example, you may not use loops or conditional statements or any Python functions that solve the problem for you.

Project Submission and Grading

You will submit your project to Gradescope. To do this, you must go to the Gradescope website for this course, find the corresponding project, and click on Submit. You must submit the following file(s) containing your solution:

- `p1.py`

Gradescope will run tests on your submission and automatically grade your solution. You will see the results of the tests and a score. If you fail any tests, you will need to go back to your solution and fix the problems. Gradescope will display the total number of points.

Notes and Hints

1. Make sure your program output matches **precisely** the output shown in the *Examples* section at the end of this document. Gradescope will not understand your submission if your output does not match exactly.
2. Although this problem can be solved using strings and string operations, to reinforce your understanding of numeric expressions and the representation of integers, you are to solve the problem using integers and integer operations only. We will study strings and string operations later in the semester.
3. You do not need to check if the user enters an integer or an integer of the proper length. That is, your program may assume that the user always enters a valid integer for this problem and nothing else. We will look at how to write more robust programs later in the semester that will handle bad input from users.
4. Remember, the integer rounding // and modulo % operators are useful for extracting the digits from an integer.
5. Decimal notation tells you how to calculate an integer if you know its digits and their positions. For example, given the digits A, B, and C, the integer represented by ABC is 100*A + 10*B + C.
6. Save your work early and often. You should save your work on some cloud-based drive (e.g., iCloud, OneDrive, Google Drive, Dropbox) to ensure that you do not lose hours of work if your Python environment crashes. You can also save your work by submitting to Gradescope often (see below). This will also allow us to have access to your work and answer any questions if you are struggling.

Questions to Consider

As you are working, consider the following questions.

1. Guessing a correct solution to this puzzle is not easy because there are so many possible guesses. How might you solve this problem to find all solutions? The answer to this includes programming constructs we have not covered. So, approach the answer to this from a human perspective. Note, for a human, this is a rather slow and tedious process.
2. What happens if a user does not enter an integer when prompted for a guess? How could you fix this problem? Note, this is not a deep question and relies on a programming construct we have not covered. So, approach the answer to this from a human perspective.

You do not need to hand in answers to these questions.

Examples

To illustrate, we show results of executing a program that meets the project specifications for two different test inputs. Everything below was produced by print statements in our solution program. The blue characters indicate the command we typed in to run our Python program. The characters in red indicate the test inputs provided as input by a human/user.

The first example run shows the result of guessing an incorrect solution; the second shows the result of guessing a correct one.

Your solution must produce the exact **identical** output, character by character. If you do not then the auto-grader may not see your solution as correct.

```
python p1.py
Solve: SLAYER + SLAYER + SLAYER = LAYERS
What is your guess? 345923
459233 == 1037769 -> False

python p1.py
Solve: SLAYER + SLAYER + SLAYER = LAYERS
What is your guess? 142857
428571 == 428571 -> True
```