Reversle

Background

There was once a boy who loved math, who aspired to be a mathematician when he grew up. At the tender age of 7, he already had a firm grasp of advanced mathematical concepts. Driven by his desire to learn, he pestered his parents to buy increasingly complicated books on mathematical concepts.

Soon, it reached a point where his entire room was filled with books, and yet his desire to learn did not slow down. His parents had purchased several wall mounted brackets (to further increase the number of books he could keep). But in no time, even those were fully stacked to the ceiling.

One evening, while reading a particularly engaging book on the "unsolvable" Collatz Conjecture, he heard a slight creaking sound coming from above him. Without further warning, the brackets holding up a large pile of books collapsed onto him, burying him under their weight. In a panic, his parents rushed into the room, navigating the labyrinth of books (now strewn all over the room), eventually digging out the boy from beneath the largest pile.

The boy was badly bruised but alive. Since that incident, he developed an irrational fear of brackets and this phobia soon spread to all things "bracket" related...

- He removed all b✱ackets in his house - choosing to purchase large shelves instead
- He could no longer watch sports matches - seeing the score b✱✱ckets reminded him of the horrible incident
- He stopped his research into income inequality - it was impossible to write without using the words "income b✱✱✱kets"

And most importantly, he found out that he could no longer handle the usage of b✱✱✱✱✱ets in mathematical formulas.

Struggling to continue his work in the field that he so loved, he decided to research alternatives to the usage of b✱✱✱✱✱ts in formulas.

After several years of research, he finally found a solution!

B✱✱✱✱✱✱s are used in the conventional way to compute as an intermediary steps before the final value is derived.

- e.g. {1 + {2 - 3}} * 4 this is a curly brace, not a b✱✱✱✱✱✱✱✱ requires 2 - 3 to be evaluated first, followed by 1 + -1, before multiplying it by 4

Therefore, he came up with a new system where equations can be represented without ✱✱✱✱✱✱✱✱, by allowing the operations to be applied in reverse

- When an operator is encountered, the previous two numbers are evaluated together with the operator, and the result is placed back into the equation
- General form: n1 n2 operator => n1 operator n2
- e.g. 2 3 - is equivalent to 2 - 3, which evaluates to -1
- e.g. {1 + {2 - 3}} * 4 can be represented as 1 2 3 - + 4 *

The next step was to increase the usage of this new system that he created. He created a set of challenges, where participants who solved it would be given his prized research, the unprecedented proof for the Collatz Conjecture - yes, he had solved it

And thus, the "challenge" called Reversle was born.

Challenge description

Objective and scoring

- The aim of the challenge is for participants to guess what the equation generated by the challenge server is within the number of attempts allowed
- Points will be scored based on the number of equations guessed correctly, and the higher the difficulty of the equation, the more points scored

Endpoint

- Expose a **POST** endpoint with the path **/reversle** on your server
- Expected input and output response content type: **application/json**

Rules

- Each equation is represented as a list of strings, each representing one token in the equation
- A total of 10 test cases will be sent to the participant's server. Each test case is sent one at a time until it is solved or the attempts allowed are exceeded
- If the guess sent by the participant's server is incorrect, the challenge server will provide more information about the correctness and the position of each token
- There is a timeout of 30 seconds for each response to be sent by the participant's server to the challenge server. If a response is not sent within this time, the current test case will be forfeited

Expected input/output format

- The first call for each test case will be sent from the challenge server to the participant's server, regarding the equation length and the attempts allowed for the current test case

```
{ "equationLength": 8, "attemptsAllowed": 10 }
```

- Participants are expected to respond with an equation of the following format:

```
{
  "equation": ["9","3","/","4","^","=","8","1"]
}
```

- Each token represents either a digit, an operator, or an equal sign
- List of valid operators are: +, -, *, /, \ {back-divide}, ^ {exponent}
    - Back divide is basically performing the division in reverse order: e.g. 3 \ 9 = 3
- Every equation can only have a single equal sign
- All digit tokens on the left side of the equal sign represents a single digit, 2 consecutive numbers are considered to be 2 seperate tokens
- Only digits are allowed on the right side of the equal sign. 2 consecutive digits would represent a 2 digit number and so-on .
- The equation on the left side of the equals must equate to the number represented on the right side
- The left side of an equation can have negative or fractional intermediate values as long as it evaluates to the right hand side eventually

- Subsequents call from the challenge server will send the equationHistory and resultHistory, to provide information about the state
- The first equation inside equationHistory corresponds with the first result inside resultHistory, and so on
- A "2" indicates that the token is both correct and in the correct position,
- A "1" indicates that the token is present in the equation, but not in the correct position
- A "0" indicates that the token is not present in the equation
- If an invalid equation is entered, the current test case will be forfeited and the server will send the next test case

Example flow - actual equation is 1 5 * 6 * = 3 0

1} Challenge server sends POST request to partipant's server

```
{ "equationLength": 8, "attemptsAllowed": 10 }
```

2} Partipant's server responds with JSON body with the guess on the equation

```
{ "equation": ["2","4","+","5","*","=","3","0"] }
```

3} Participant's equation is incorrect, thus challenge server responds with the equation length, equation and result history, and attempts left

```
{
    "equationLength": 8,
    "equationHistory": [
        ["2","4","+","5","*","=","3","0"],
    ],
    "resultHistory": [
        ["0","0","0","1","2","2","2","2"],
    ],
    "attemptsLeft": 9
}
```

4} Participant's server sends another response to the challenge server and gets the equation correct

```
{ "equation": ["1","5","*","6","*","=","3","0"] }
```

5) Challenge server sends details about the next test case, or stops sending if no test cases left

```
{ "equationLength": 10, "attemptsAllowed": 20 }
```

Hints

- Use \\ in the JSON response to represent back divide as \ is an escape character

Challenge created by Dion & Moses

Contact us at dion.ang@credit-suisse.com or moses.yong@credit-suisse.com_