Digital Root Calculation

Give notes for this problem. Given an integer num, repeatedly add all its digits until the result has only one digit, and return it. Here my solution is if (num%9==0) then return 9. else return num%9

2/2

This problem is about finding the "digital root" of a number, which involves repeatedly summing the digits of a number until a single digit remains. The solution leverages a mathematical property to achieve this efficiently.

Problem Explanation

1. Given a number `num`, the goal is to repeatedly sum its digits until only one digit remains.
   - Example: `38 → 3 + 8 = 11 → 1 + 1 = 2`
   - Output: `2`
2. A naive approach involves repeatedly summing digits in a loop until the number is a single digit. However, this can be computationally expensive for large numbers.
3. The optimized solution involves modular arithmetic based on the concept of "digital root."

Concept of Digital Root

- The digital root of a number is the single digit obtained by repeatedly summing its digits.
- It can be computed efficiently using modular arithmetic:
  - **Digital Root Formula:** For a number n,

$$n \mod 9$$

    gives the digital root, except:

- If $n \mod 9 = 0$ and $n > 0$, the result is 9 instead of 0.

Key Observations

1. A number and the sum of its digits are congruent modulo 9.
   - Example: 38
     $38 \mod 9 = 2$
     Sum of digits: $3 + 8 = 11, 11 \mod 9 = 2$
2. The repetitive summation of digits reduces to the modulo operation because:
   - After each iteration, the number is effectively reduced to $num \mod 9$.

Optimized Solution

The solution avoids the iterative digit summation by directly using the modulo operation:

- **Steps:**
  1. If `num == 0`, return `0` (edge case).
  2. If `num % 9 == 0` and `num > 0`, return `9`.
  3. Otherwise, return `num % 9`.
- **Code Logic (Pseudocode):**

```java
if (num == 0) {
    return 0;
} else if (num % 9 == 0) {
    return 9;
} else {
    return num % 9;
}
```

Edge Cases

1. **Zero Input:**
   - $num = 0 \rightarrow$ Return 0.
2. **Multiples of 9:**
   - $num = 18$, $num = 27$, $num = 9 \rightarrow$ Return 9.

Why the Formula Works

- This property relies on modular arithmetic:
  - Each digit d contributes $d \times 10^k$ to the number, where k is the position.
  - $10^k \equiv 1 \mod 9$, so the contribution modulo 9 is just d.
  - Hence, the number modulo 9 equals the sum of its digits modulo 9.

Time Complexity

- $O(1)$: The solution involves a constant-time modulo operation, making it highly efficient.

Space Complexity

- $O(1)$: No extra space is used.

Examples

1. $num = 38$
   $38 \mod 9 = 2 \rightarrow$ Return 2.

2. $num = 12345$
   Sum digits: $1 + 2 + 3 + 4 + 5 = 15 \rightarrow$ Sum again: $1 + 5 = 6$
   $12345 \mod 9 = 6 \rightarrow$ Return 6.
3. $num = 9$
   $9 \mod 9 = 0$, but since $num > 0$, Return 9.

Would you like me to provide additional examples or test cases?

1/2