Complete Documentation

Intuition

The problem focuses on counting numbers with unique digits in the range $0 \leq x < 10^n$. The key observation here is that a number is considered valid only if no digit repeats within it. For smaller values of n, this is straightforward to compute. However, as n increases, the possible combinations of unique digits require a systematic approach. By leveraging the constraints of unique digit selection, we can use a combinatorial formula to calculate the count efficiently.

Approach

We break the problem into smaller cases. For $n=0$, the only valid number is 0. For $n=1$, all single-digit numbers (0–9) are valid, resulting in 10 numbers. For $n=2$, the first digit can be chosen from 1–9, and the second digit from 0–9 excluding the first. This results in $9\times9=81$valid numbers. For $n=3$, the first digit has 9 options, the second has 9, and the third has 8, giving $9\times9\times8=648$ unique numbers. This pattern generalizes for higher values of n, where the choices for each digit decrease as n increases.

Recursive Formula

The formula to compute the count of unique numbers for $n=k$ is derived from the constraints on digit selection. For $k=1$, the count is 10 (all single-digit numbers). For $k=2$, the count is 9×9 , and for $k=3$, it is $9\times9\times8$. Extending this pattern, the count for $n=k$ is:

$$f(k)=9\times9\times8\times\cdots\times(10-k+1),$$

where the first digit has 9 choices (to exclude 0), and subsequent digits have decreasing choices to ensure uniqueness. The total for n is the summation of all f(k) from $k=0$ to n.

Implementation Insight

The solution iteratively computes the count for each digit length from 1 to n. It starts with a base case for n=0, where the total count is 1. For each k from 1 to n, the algorithm calculates the number of unique numbers by iterating over the available digits, reducing the number of choices for each position. This approach ensures that the computation is efficient and avoids unnecessary recalculations.

Example Walkthrough

For n=2, the valid numbers are those between 0 and 9 with no repeated digits. For n=1n = 1, there are 10 numbers (0–9). For n=2, the first digit has 9 choices (1–9), and the second digit has 9 options (0–9 excluding the first digit), resulting in 81. Adding these, the total count is 91. Similarly, for n=3, we extend the logic by adding a third digit with 8 choices, giving 648 additional numbers.

Complexity Analysis

The time complexity is $O(n^2)$. The outer loop iterates over each digit length from 1 to n, while the inner loop computes the product of available digits for that length. Space complexity is O(1) since no additional data structures are used; the computation is done in place. This makes the solution efficient for the given constraint of $0 \leq n \leq 8$

Applications

This problem has applications in scenarios where constraints on unique digits or characters are essential, such as generating secure PIN codes, license plates, or unique identifiers. Understanding combinatorics and constraints in such cases helps in efficient algorithm design. The approach can also be extended to problems involving unique combinations or permutations under specific conditions.