Perpetua Sarina Senatus

An algorithm for solving the Car Talk Puzzle problem (Python)

Design an algorithm that takes as input a 9-digit number where no digit appears twice and produces as output an arrangement of the same nine digits corresponding to the next highest number. If no such number exists, the algorithm should indicate this. So, for example, if the input is 781623954, the output would be 781624359.

- **Step 1- input a 9-digit number**

The algorithm is a 9-digit number with non-repeating digits.

Example : n = 123456789

- **Step 2 - add one to get new_n initialized**

new_n = n +1 = 123456789 + 1 → 123456790

We can start to check each number by adding 1 to new_n

- **Step 3 – a loop with a 1 billion limit**

While new_n <1000000000 (no numbers exceed 9 digits)

This loop ensures that every new_n to check will be less than 1 billion.

- **Step 4 - convert n and new_n to string**

n_str = str(n) → so 123456789 is now "123456789"

new_n = str (new_n) → so 123456790 is now "123456790"

We convert them to strings to make it easy to compare the digits.

- **Step 5 – digits comparison**

We use sets to compare, ensure no duplicates, and verify all original digits' presence.

Set of n (1, 2, 3, 4, 5, 6, 7, 8, 9) **AND** Set of new_n **(**1, 2, 3, 4, 5, 6, 7, 9, 0): are not equal.

- **Step 6 - return**

If the digits match and there are no duplicates, return new_n.

- **Step 7 – if step 6 failed, increment by 1**

new_n + 1 = 123456790 + 1 → 123456791

- **Step 8 – Repeating**

Repeat step 5 for digits comparison

Set of n (1, 2, 3, 4, 5, 6, 7, 8, 9) **AND** Set of new_n **(**1, 2, 3, 4, 5, 6, 7, 9, 1): 1 appears twice, not equal

- **Step 9 – Repeating**

Repeat step 7 to increment by 1 :

new_n + 1 = 123456791 + 1 → 123456792

Repeat step 5 for digits comparison:

Set of n (1, 2, 3, 4, 5, 6, 7, 8, 9) **AND** Set of new_n **(**1, 2, 3, 4, 5, 6, 7, 9, 2): 2 appears twice, not equal

- **Step 10 – Repeating**

Repeat step 7 to increment by 1 :

new_n + 1 = 123456792 + 1 →123456793

Repeat step 5 for digits comparison:

Set of n (1, 2, 3, 4, 5, 6, 7, 8, 9) **AND** Set of new_n **(**1, 2, 3, 4, 5, 6, 7, 9, 3): 3 appears twice, not equal

- **Step 11 – Repeating**

Repeat step 7 to increment by 1:

new_n + 1 = 123456793 + 1 → 123456794

Repeat step 5 for digits comparison:

Set of n (1, 2, 3, 4, 5, 6, 7, 8, 9) **AND** Set of new_n **(**1, 2, 3, 4, 5, 6, 7, 9, 4): 4 appears twice, not equal

- **Step 12 – Repeating**

Repeat step 7 to incrementing by 1:

new_n + 1 = 123456794 + 1 → 123456795

Repeat step 5 for digits comparison:

Set of n (1, 2, 3, 4, 5, 6, 7, 8, 9) **AND** Set of new_n **(**1, 2, 3, 4, 5, 6, 7, 9, 5): 5 appears twice, not equal

- **Step 13 – Repeating**

Repeat step 7 to incrementing by 1 :

new_n + 1 = 123456795 + 1 →123456796

Repeat step 5 for digits comparison:

Set of n (1, 2, 3, 4, 5, 6, 7, 8, 9) **AND** Set of new_n **(**1, 2, 3, 4, 5, 6, 7, 9, 6): 6 appears twice, not equal

- **Step 14 – Repeating**

Repeat step 7 to incrementing by 1:

new_n + 1 = 123456796 + 1 → 123456797

Repeat step 5 for digits comparison:

Set of n (1, 2, 3, 4, 5, 6, 7, 8, 9) **AND** Set of new_n **(**1, 2, 3, 4, 5, 6, 7, 9, 7): 7 appears twice, not equal

- **Step 15 – Repeating**

Repeat step 7 to increment by 1:

new_n + 1 = 123456797 + 1 → 123456798

Repeat step 5 for digits comparison:

Set of n (1, 2, 3, 4, 5, 6, 7, 8, 9) **AND** Set of new_n **(**1, 2, 3, 4, 5, 6, 7, 9, 8): are equal

Return new_n → 123456798 (step 6)

- **Step 16 - end of Condition**

If new_n exceeds 999999999 without finding equal sets, the output would be " no more possibilities"