Project_Euler_024

February 4, 2018

1 Project Euler Problem 24

A permutation is an ordered arrangement of objects. For example, 3124 is one possible permutation of the digits 1, 2, 3 and 4. If all of the permutations are listed numerically or alphabetically, we call it lexicographic order. The lexicographic permutations of 0, 1 and 2 are:

```
012   021   102   120   201   210
```

What is the millionth lexicographic permutation of the digits 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9?

```
In [4]: # I hadn't touched this problem in a while, so looking back at it
        # again, it took me some time to tease out the logic of what I was
        # doing.

        # When going through permutations of n elements in lexicographic
        # order, we don't change the first element until we've exhausted
        # all (n-1)! permutations of the remaining elements to the right.
        # This logic can be applied going to the right as our set of
        # remaining elements shrinks.

        # We know that the first permutation is 0123456789, and we want
        # to move ahead 999999 steps.  To figure out the first digit of
        # the millionth permutation, we see how often the remaining
        # 9 digits get shuffled through all 9! permutations.
        # 9! = 362880, and 999999/362880 is a little over 2,
        # so we have to remove element 2 (using Python numbering
        # starting at 0) from our originalList
        # and place it at the front of our solutionList.  There are
        # 999999 - 2*9! = 274239 permutations remaining.

        # Move onto the second digit.  We have to see how many permutations
        # happen in the rightmost 8 digits to see how many times
        # we have to change the second digit.  8! = 40320, and
        # 40320 goes into 274239 a little over 6 times, so we pick
        # element 6 from the remaining originalList, which is 7.
        # Our originalList is now missing the numbers 2 and 7.
        # There are 274239 - 6*8! = 32319 permutations remaining.
```

```python
# Keep going until you get to the end.

from math import floor, factorial

originList = [0,1,2,3,4,5,6,7,8,9]
count = 10**6-1
solutionList = [0,0,0,0,0,0,0,0,0,0]

for i in range(10):
    solutionList[i] = originList[int(floor(count/factorial(9-i)))]
    count = count % factorial(9-i)
    originList.remove(solutionList[i])

print("The millionth permutation is {}."
     .format(int(''.join(list(map(str, solutionList))))))
```

```
The millionth permutation is 2783915460.
```