Problem24

April 16, 2022

1 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?

```
[ ]: digit_seq = [9, 8, 7, 6, 5, 4, 3, 2, 1, 0]

     pc = 1 #permutation count
     pl = 1000000 #permuation limit
```

Try an iterative function… but this hits *'maximum recursion depth'* for longer sequences..

```
[ ]: def next_perm(seq, pos):
         global pc
         global pl
         l = len(seq)
         cd = seq[pos] #current digit
         ts = [] #temp sequence
         td = 0 #temp digit


         for i in range(0, pos):
             ts.append(seq[i])
         ts.sort()
         x = 0
         while x < len(ts) - 1 and ts[x] <= cd:
             x = x + 1
         if ts[x] <= cd:
             if pos < l - 1:
                 next_perm(seq, pos + 1)
             return
         else:
             pc = pc + 1
             td = ts.pop(x)
```

```
        ts.append(cd)
        ts.sort()
        ts.reverse()
        ts.append(td)
        for i,d in enumerate(ts):
            seq[i] = d
        if pc >= pl:
            print("seq =", seq, " ts = ", ts, " cd = ", cd)
            return

        next_perm(seq, 1)
        return
    return
next_perm(digit_seq, 1)
print("perms = ", pc)
```

Try just increasing the number by 1 then checking whether the list of numbers is the same... Works but takes about 10s to find the 6000th permutation and 1m 35s to find the 50000th so on course for around 30min for 1 x 10^6

[ ]:
```
def find_perms(seq):
    print(seq)
    global pc
    global pl
    l = len(seq)
    original_seq = seq.copy()
    original_seq.sort()

    while pc < pl:

        i = 0
        inc = 1

        while inc == 1 and i < l:
            seq[i] = seq[i] + inc
            if seq[i] == 10:
                seq[i] = 0
                inc = 1
            else:
                inc = 0
            i = i + 1

        new_seq = seq.copy()
        new_seq.sort()

        if new_seq == original_seq:
            pc = pc + 1
    print(seq)
    return

find_perms(digit_seq)
```

Try re-writing the iterative method as a loop...

[ ]:
```
def perm_loop(seq):
    global pc
    global pl
    l = len(seq)
    ts = [] #temp sequence
    td = 0 #temp digit
    n = 1

    while n < l and pc < pl:
        cd = seq[n]
        ts = seq[0:n].copy()
        ts.sort()
        n1 = 0
        l1 = len(ts)
        change = False
        while n1 < l1 and change == False:
            if ts[n1] > cd:
                change = True
                td = ts.pop(n1)
                ts.append(cd)
                ts.sort()
                ts.reverse()
                ts.append(td)
            n1 = n1 + 1
        if change == True:
            for i in range(0, len(ts)):
                seq[i] = ts[i]
            n = 1
            pc = pc + 1

        else:
            n = n + 1

    print(seq , " count = ", pc)


perm_loop(digit_seq)
```

```
[0, 6, 4, 5, 1, 9, 3, 8, 7, 2]  count =  1000000
```

It works! 2.4s to find the answer. Note that it gives the numbers in reverse order, so actual answer is 2783915460