MA/CSSE 473 – Design and Analysis of Algorithms   In-class Exercises   Day 13 Fall, 2014

4. Which permutation follows each of these in lexicographic order?

183647520 **183650247**   471638520 **471650238**

5. Try to write an algorithm for generating the next permutation, with only the current permutation as input.

```
def next(self):
        "return current permutation and calculate next one"
        if not self.more:
            return False
        returnValue = list(self.current)
        i = self.n - 2
        while self.current[i] > self.current[i + 1]:
            i -= 1 # This avoids array-out-of-bounds because
        if i == - 1: # in Python, a[-1] means a[len(a)-1]
            self.more = False
        else:
            j = self.n - 1
            while self.current[i] > self.current[j]:
                j -= 1
            self.swap(i, j)
            self.reverse(i + 1, self.n - 1)
        return "".join([str(v) for v in returnValue])
```

6. If the lexicographic permutations of the numbers [0, 1, 2, 3, 4] are numbered starting with 0, what is the number of the permutation 14023? How do you get this?

1 * 4! = 24, then (decrease-and-conquer) look at 3021
3 * 3! = 18, then look at 021
0 * 2! = 0, then look at 01
0 * 1! = 0
24+18 = 42

7. Write an algorithm which, given a permutation of the numbers 0..n-1, calculates its (zero-based) position in the lexicographic ordering of all of the permutations of 0..n-1.

```
def permNumber(p):
    """assumes that p is a permutation of 0..n-1.
    returns k such that p is the kth lexicographic
    permutation of those numbers."""
    p = [int(i) for i in p] # make a list of ints
    n = len(p)              # so we can do arithmetic
    factList = [ft.get(i) for i in range (n-1,-1,-1)]
    sum = 0
    for i in range(n):
        sum += p[i] * factList[i]
        for j in range(i + 1, n):
            if p[j] > p[i]:
                p[j] -= 1
    return sum
```

8. In the lexicographic ordering of permutations of [0, 1, 2, 3, 4, 5], which permutation is number 541? How do you get this?

p = [0, 1, 2, 3, 4, 5] 5! = 120. 541 = 4(120) + 61. First number in the permutation is p[4] = 4.
p = [0, 1, 2, 3, 5] 4! = 24 61 = 2(24) + 13. Next number in permutation is p[2] = 2.
p = [0, 1, 3, 5] 3! = 6 13 = 2(6) + 1. Next number in permutation is p[2] = 3.
p = [0, 1, 5] 1 = 0(2) + 1. Next number in permutation is p[0] = 0.
P = [1, 5] 1 = 1(1) + 0

Code:

```
def kthPermutation(s, k):
    """return kth lexicographic permutation of elements in list s. Inverse of permNumber()"""
    s = list(s)
    result = []
    factTable = [ft.get(i) for i in range (len(s)-1,-1,-1)]
    for divisor in factTable:
        multiple = k // divisor
        k = k % divisor
        element = s[multiple]
        result.append(element)
        s.remove(element)
    return result
```