MATH 343, SPRING 2013, ASSIGNMENT 6

DUE THURSDAY MARCH 21, 2013 IN CLASS

Do **any three** of the following four problems. If you do more than three, only the first three will be graded.

(1) The following questions concern the Reflected Binary Code (RBC) which yields a Gray code for binary strings. Let $R(n)$ be the sequence generated for binary strings of length n. Let $r_n(k)$ denote the k-th element of the sequence $R(n)$.
 (a) Find $r_9(267)$.
 (b) Which word follows 111100011010 in $R(12)$?
 (c) Find a formula for k such that $r_n(k) = 1^n$.

(2) (a) For k-subsets in revolving door order prove that the following formula holds

$$\operatorname{rank}(T) = \begin{cases} \sum_{i=1}^{k}(-1)^{k-i}\binom{t_i}{i} & \text{if } k \text{ is even} \\ \sum_{i=1}^{k}(-1)^{k-i}\binom{t_i}{i} - 1 & \text{if } k \text{ is odd} \end{cases}$$

where $T = (t_1 < \cdots < t_k)$ is a k-subset written as a list in increasing order.
 (b) Use the previous part to write an algorithm (pseudocode is fine) for rank and unrank in the revolving door order.

(3) A permutation is *even* if it can be transformed into the identity permutation with an even number of transpositions. A permutation is *odd* if it is not even. (You may know considerably more about this from other classes)
 (a) Prove that the parity of a permutation is the same as the parity of its rank in Trotter-Johnson order.
 (b) Prove that the algorithm given as Algorithm 2.19 on the second page calculates the parity of a permutation.

(4) (a) Calculate the runtime of Algorithm 2.20 on the second page. Don't assume that Algorithm 2.19 is constant – calculate its runtime too.
 (b) Step through the algorithm on some examples of different permutations. Choose examples so that it is convincing that the algorithm works correctly. Make sure at least one of your examples has size more than 3.

This is from page 63 of "Combinatorial Algorithms" by Kreher and Stinson.

```
Algorithm 2.19: PERMPARITY (n, π)

for i ← 1 to n do a[i] ← 0
c ← 0
for j ← 1 to n
     ⎧ if a[j] = 0
     ⎪          ⎧ c ← c + 1
  do ⎨          ⎪ a[j] ← 1
     ⎪   then   ⎨ i ← j
     ⎪          ⎪ while π[i] ≠ j do ⎧ i ← π[i]
     ⎩          ⎩                   ⎩ a[i] ← 1
return ((n − c) mod 2)
```

```
Algorithm 2.20: TROTTERJOHNSONSUCCESSOR (n, π)

external PERMPARITY()
st ← 0
for i ← 1 to n do ρ[i] ← π[i]
done ← false
m ← n
while m > 1 and not done
     ⎧ d ← 1
     ⎪ while ρ[d] ≠ m do d ← d + 1
     ⎪ for i ← d to m − 1 do ρ[i] ← ρ[i + 1]
     ⎪ par ← PERMPARITY(m − 1, ρ)
     ⎪ if par = 1
     ⎪          ⎧ if d = m
     ⎪          ⎪   then m ← m − 1
     ⎪   then   ⎨          ⎧ temp ← π[st + d]
     ⎪          ⎪   else   ⎨ π[st + d] ← π[st + d + 1]
  do ⎨          ⎪          ⎪ π[st + d + 1] ← temp
     ⎪          ⎩          ⎩ done ← true
     ⎪          ⎧ if d = 1
     ⎪          ⎪   then   ⎧ m ← m − 1
     ⎪          ⎪          ⎩ st ← st + 1
     ⎪   else   ⎨          ⎧ temp ← π[st + d]
     ⎪          ⎪   else   ⎨ π[st + d] ← π[st + d − 1]
     ⎪          ⎪          ⎪ π[st + d − 1] ← temp
     ⎩          ⎩          ⎩ done ← true
if m = 1
  then return ("undefined")
```