CSCI 570 - HW 2

1. Given two sets A and B with n positive numbers respectively.
   Let the set A be ai, and sorted i.e $a_1>a_2>a_3>\ldots>a_i$
   B be bi, $b_1>b_2>b_3>\ldots>b_i$
   Let S be the optimal solution
   Then a_1 is to be paired b_i and a_i with b_1
   Let S' be another optimal solution
   Then a_1 is paired with b_1 and a_i with b_i
   Now, Payoff(S)/Payoff(S') = (TTs a_i^b_i)/(TTs' a_i^b_i)
   = [(a_1^b_i)*(a_i^b_1)]/ [(a_1^b_1)*(a_i^b_i)]
   = (a_1/$_i$)^(b_i-b_1)
   Since b1>bi, [Payoff(S)/Payoff(S')]<1, This contradicts that S is optimal solution, therefore a_1 should be paired with b_1
   Time Complexity:
   If the elements are not sorted then the complexity will be O(nlogn)
   If the elements are already in a sorted order then the complexity is O(n)

2. Greedy algorithm can be the most efficient algorithm to determine at which gas station one should stop, this is because greedy algorithm is mostly used for optimization problems. This will help to make the optimal choice to stop at a particular gas gi station after di distance

   Let $(g_1,g_2,g_3,\ldots,g_n)$ be the gas stations
   Let $(d_1,d_2,d_3,\ldots,d_m)$ be the optimal solutions

   Now we must travel as far as possible before stopping for a refill. Suppose if we start at USC and predict to refill at i+1th station, if not then at ith gas station
   If we are unable to reach g_1, since the distance between each gas station is p and when the tank is full the car can travel p miles(distance between two neighboring gas stations), $d_1<=g_1$ (the car may also need to be filled before if the tank is not full). Assume the case where the car tank is not filled completely, then $d_1<g_1$, now replace the first stop with g_1, the new solution is $(g_1, d_2,d_3,\ldots,d_m)$
   Assume the above as case for 'a' number of time then the new optimal solution will be $(g_1,g_2,\ldots,g_{a-1},d_a,d_{a+1},\ldots,d_m)$
   Induction Hypothesis: assume $d_a<=g_a$, the replace d_a with g_a
   By induction $d_m<=g_m$, thus the optimal solution will be
   $(g_1,g_2,g_3,\ldots,g_m)$
   The time complexity of the algorithm will be O(m), since there are m gas station refills in the above solution

3. Let S' be the sequence of events
   Let S be the sub-sequence of events

   Pseudo Code:
   While i<n:

```
        While j<m:
            If S'[j]==S[i]:
                Count = count+1
                Total = total+1
            Else:
                Total+1
                Continue
    If total>=count:
        Print("Yes")
```

Time complexity:
The time complexity of the outer loop will be O(n)
The time complexity of the inner loop will be O(m)
Total time complexity = O(n) + O(m)
= O(n+m)
The algorithm prints "Yes" if S is a sub-sequence of S'

Proof:
Let A be the sequence of events ($a_1,a_2,\ldots,a_n$) found to be true by greedy algorithm
Let B be the sequence of events ($b_1,b_2,\ldots,b_m$) found to be true and also is a optimal solution

Now we need to prove using induction that the solution returned by the greedy algorithm is an optimal solution
$a_n>=b_m$, i.e the set solutions returned by greedy algorithm will belong to optimal solution
Case 1: n=1 and m=1, only a single element
$a_1>=b_1$, only a single event, if they are same then the solution returned by the greedy algorithm will be the optimal solution
Case 2: n>1 and m>1, n-1<n and m-1<m
By induction hypothesis n-1 and m-1 belong to the sequence A and B and hence an>bm, so the greedy method also gives an optimal solution

4. Assume that the greedy algorithm solution is sequence of n packages
Ith package will have a weight w_i and will be loaded in truck t_i
Assume 'a' trucks are used to load n packages
i.e, $t_n=a$
Assume greedy algorithm outputs an optimal solution and also non-optimal solution for the sequence of packages
$\forall$ i<k, $t_i=t'_i$ (is optimal solution)
$T_k!=t'_k$ (is non-optimal solution)

Case1: $t_k>t'_k$, this is the case when some of the trucks are not filled completely and the skipped to fill the other trucks, however according to greedy algorithm, we cannot switch to another truck without completely filling the current truck, this case violates the greedy algorithm, therefore this case cannot be included in the sequence of solutions

Case2:$t_k<t'_k$, in this case the truck will be switched according to the optimal solution.

In this case the truck t_k will be switched to t_{k+1} only after filling the truck t_k, this is according to the greedy algorithm, because the truck t_k will be switched to truck t_{k+1} only after the t_k is fully switched
Therefore the greedy solution is an optimal solution

5. (i) T(n) = 4T(n/2) + (n^2)logn
   a=4, b=2, k=2, p=1
   $\log_b a$= 2
   ->since $\log_b a$ = k
   p>-1
   Therefore: Ø((n^2)*(logn)^2)

(ii) T(n) = 8T(n/6) + nlogn
a=8, b=6, k=1, p=1
$\log_b a$= $\log_3 4$ =1.2
->since $\log_b a > k$
Therefore: Ø(n^($\log_6 8$)))

(iii) T(n) = sqrt(6006)T(n/2) + (n^sqrt(6006))
T(n) = 77.5T(n/2) + (n^77.5)
a=77.5, b=2, k=77.5, p=0
$\log_b a$ = $\log_2 77.5$= 6.3
->since $\log_b a < k$
p>=0
Therefore: Ø(n^sqrt(6006))

(iv) T(n) = 10T(n/2) + (2^n)
T(n) = 10T(n/2) + logn
a=10, b=2, k=0, p=1
$\log_b a$ = $\log_2 10$ = 3.3
->since $\log_b a > k$
Therefore: Ø(n^($\log_2 10$))

(v)T(n) = 2T(sqrt(n)) + logn
Substitute: m=logn
T(2^m) = 2T(2^(m/2))+m
Substitute: s=2^m
T(s) = 2T(s/2)+logs
a=2, b=2, k=0, p=1
$\log_b a$ = $\log_2 2$ =1
->since $\log_b a > k$
Therefore: Ø(s^$\log_2 2$)
=>Ø(2^m)
=>Ø(2^logn)