Homework-3

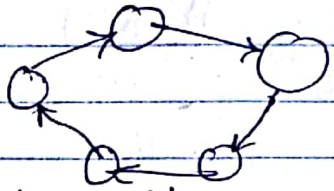
(1) Read. ✓

(2) Since all the streets are made one way, let us consider it as directed graph.

→ As per the mayor's claim, there is a way to drive from any one intersection to other legally.

Let intersections be denoted using vertices.

eg:



So this is a ~~strongly~~ connected graph.

Any intersection can be reached from any other intersection directly or via other intersections.

(a) 1. Choose any node 'a' from the graph G.
Then run BFS on node 'a', until all nodes are visited ~~your reach back to 'a'~~.

~~(b)~~ 2. Do the same step-1 by reversing the Graph

3. If all the nodes are visited in the graph G and its inverse, then it is a ~~strongly~~ connected graph.

→ So the complexity for m edges and n nodes here will be $O(m+n)$.

(b) Here, let town hall ~~f~~ be the initial node.

1. Run BFS from town Hall in the graph.
2. Run BFS in the reverse direction (inverted graph).
3. If all the nodes are visited in the graph & its inverse and if every node can reach town hall, then it is a strongly connected graph.

(3) If there exists at least one vertex with indegree 0, then the graph is DAG.

→ Algorithm:

```
Check if there is a node with no incoming edges.
    If it is true then
        Choose that node {v} and order it first
        Delete that node from graph G (& its outcoming edges)
        Recursively compute a topological ordering of G-{v}
        and append this order after v.
    Else If there exists no node with no incoming edges
        then Cycle exists and its not DAG.
```

→ The running time of this algorithm for m edges an n vertices is $O(m+n)$.

(4) Assuming to go with full gas tank from USC.

```
while(the current position is not Santa Monica && distance of ≤ p miles from current posit is not Santa Monica)
    If there is a gas stop at p miles from current position (distance) then
            fill gas tank at that stop
    Else if there is no gas stop at p miltes from
        current position then
            fill gas tank at the gas stop right
            before p miles distance.
    EndIf
    The new position (gas station) will become
    the current position
EndWhile.
```

⑤ Let us assume that the optimal solution includes $\{b_1, b_2, b_3, \ldots, b_n, b_{n+1}, b_{n+2}\}$ boxes.

And the greedy solution inculdes $\{b_1, b_2, \ldots b_n\}$ boxes.

→ For the assumed optimal solution the boxes b_{n+1} and b_{n+2} (~~will~~ might exceed the weight limit of the truck) and will not fit in the truck leaving the solution $\{b_1, b_2, \ldots b_n\}$ as the optimal one.

→ If we decrease the number of boxes per truck say $\{b_1, b_2, b_3 \ldots b_{n-1}\}$ and trying to make the next truck more compactly packed. In that case instead of $\{b_{n+1}, b_{n+2}, \ldots b_{n+x}\}$ boxes arrangement in the next truck, there will be extra b_n th box which will not fit the truck. So based on that, the current solution is the optimal one. The new one might take more number of trucks.

→ Hence it is proved that the greedy algorithm uses the fewest possible trucks and hence, "stays ahead" of any other solution.

⑥ 1. We rearrange both the sets in non-decreasing order.

2. Then we implement $\prod_{i=1}^{n} a_i^{b_i}$, where n is the total number of elements $(n(A) = n(B))$, a_i is the i^{th} element of set A and b_i is the ith element of set B.

→ In this way, the value of $a_i^{b_i}$ will keep on increasing with i, and after multiplying all answers we can get the maximum value possible. (i.e. maximum payoff).

→ The running time for sorting will be $O(n \log n)$ and then for calculating payoff will be $O(n)$.

→ Hence, the overall complexity will be $O(n \log n)$.

(7) Let us consider a sequence S and its subsequence S'. Let S have i elements and S' have j elements. (S_i be ith element of S and S'_j be jth element of S')

```
First i=j=1
~~I~~ while (last element of S' has not found a match in S
        and S has not reached its last element)
    If $S'_j == S_i$ then
        $S'_j$ found its match; Store i in $m_j$
        i++;
        j++;
    Else If $S'_j != S_i$ then
        just check next value in S i.e. i++
~~Els~~ EndIf
EndWhile
If S' has reached its last element and crossed
  it i.e. (j+1)th element, then
    return the subsequence $m_1, m_2, \ldots m_j$
(~~Else return~~ "S' ~~is not the subsequence of S~~".)
End If.
```

(8) (a) → At most we can use 4 pennies as 5 pennies can be replaced by a nickel.

→ At most we can use 1 nickel as 2 nickels can be replaced by a dime.

→ At most we can use 2 dimes as value like 30 cents can be replaced by a quarter and a nickel, and soon.

→ Also in case of 2 dimes and one nickel, we can use one quarter.

(b) Let us consider a set of {Pennies, dimes, quarters}
If we want change of 8 cents, here we will use 8 pennies i.e. 8x1 ⇒ $\boxed{8}$ coins.

Instead if we had all and used greedy strategy then 1 nickel + 3 pennies ⇒ $\boxed{4}$ coins

So the given set does not give an optimal solution.

Scanned by CamScanner