For the first step, the algorithm will store node A into clique C, since A has the biggest degree ($d_A=6$), then node Ⓗ & Ⓘ will be removed from the network, since they are not adjacent to Ⓐ

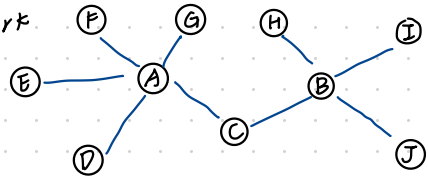
For step 2, only one of the remaining nodes will be added to (C), since Ⓗ, Ⓘ has been removed, the degree of the remaining nodes are all "1", for which they are all connected to Ⓐ; however by step 2, if we add one more node to (C), all the other nodes will be removed from the network afterwards, which means that the max Clique being return from the algorithm will be a clique with a size of "2".

But the actual maximum clique for the network should be $C=\{B,H,I\}$, $|C|=3$

Suppose we have V vertixes in the network, then the worst case run time complexity will be $O(|V|^2)$ → ① complexity of checking adjacency nodes: $O(|V|^2)$
$G(V,E)$ ② Remove the nodes that aren't adjacent: $O(V)$

(b)
The proposed greedy algorithm might not be able to find a maximum clique

consider the network



the maximum clique size is "2"
(any clique which contains Ⓐ and any nodes connected to it, or any nodes that connected to Ⓑ)

However, by the proposed algorithm, all the nodes will be removed and Ⓒ will be the only node remaining in the network → not able to find a clique

the time complexity will be $O(|V|^2)$ ← $O((V)(V-1))$
$G(V,E)$

(c)
The proposed algorithm is likely to find a maximum clique, since the algorithm starts from clique size, $k=2$, which means every pair of nodes that has been connected with an edge, the algorithm further starts to search for the existing connection between each pair of nodes. By the procedure, the algorithm is able to check all the nodes and the relationships of them. The algorithm is able to return and find the maximum Clique
The time complexity: $O(n^n)$
- Complexity for searching clique with size k: $C_k^n \to n^k$
- largest k could be up to "n" (# of nodes in the graph)

Q4

(a)

The proposed greedy algorithm might not be able to find a minimum clique.

Consider the network