//For grader: question (b): The thorough explanation of the algo can
//be found in the description of proof of correctness. And the bipartite
//graph doesn't contain every detail, and only serves as a
//concept drawing, please refer to the algo for more info about the graph.

Question (a):

Counterexample:

Supposed we have:

1) $n * n$ Matrix C where $n = 6$
2) Peg number $k = 2$

Description of discs:

1) Disc 1 is the biggest disc
2) Disc 2 can be placed on Disc $1, 3, 4, 5, 6$
3) Disc 3 can be placed on Disc 1
4) Disc 4 can be placed on Disc 1
5) Disc 5 can be placed on Disc $1, 3, 4$
6) Disc 6 can be placed on Disc $1, 4$

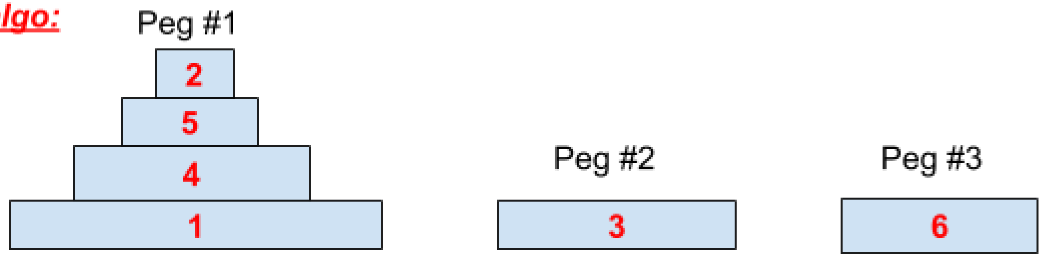
Matrix C:

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \end{matrix}$$

All the discs can be put onto 2 pegs as below:

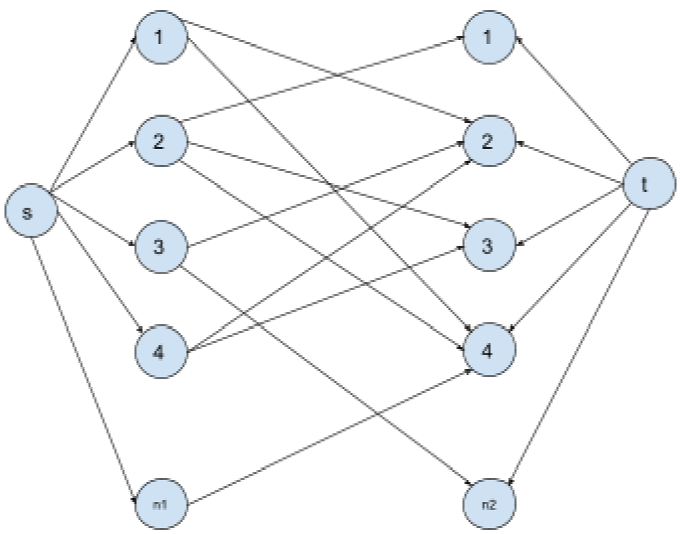
However, using the greedy algo, we first find the largest number of discs that can be put onto one peg and then remove these discs from the list D. This will result as below:

Greedy algo:



The algo will generate 3 pegs since disc3 and disc4 cannot be put onto the same peg, which is apparently a wrong result.

Question (b):



*This is a just a concept drawing to illustrate my thinking

Algorithm to solve AlgoRU's stacking puzzle

Some general description:
This algo attempts to find stacking of discs with k or fewer pegs. We want to place $n-k$ discs on top of other discs, and each time we put a disc on top of another, we are actually decreasing the total number of pegs by 1. Any matching M of G will result in $n-|M|$ pegs. So to achieve at most k pegs, there needs to exist a matching M of G with at least $n-k$ edges

1. Construct a bipartite graph G which will have $2n$ vertices with $(n1\ (left\ part) = n2\ (right\ part)\ = n\ )$
2. Assign the capacity of any edge e from s to any vertex v on the left part to 1
3. Assign the capacity of any edge e from s to any vertex v on the right part to 1 as well
4. Assign the capacity of any edge e from the left part to right part to 1

5. There exists an edge from any vertex $v1 \in n1$ to any vertex $v2 \in n2$ if $v1$ can be put on top of $v2$
6. Find maximum matching Max of G, and according to the rationale above:
   1) If $|Max| \geq n - k$, then it is possible to put n discs on k pegs → print "YES"
   2) If $|Max| \leq n - k$, then it is impossible to put n discs on k pegs → print "NO"

Proof of Correctness:

As illustrated and explained in the algo that if we want to put n discs on k pegs, then we need to put at least $n - k$ discs on top of other discs. This means that there needs to be at least $n - k$ matchings in the graph. For example, assume we have n discs that are initially put on n pegs. When we observe an edge e that is directed from $n1$ to $n2$, we know that there are at least 2 discs that can be put together, so the peg count is decremented by 1. And for the same reason, we would decrement the peg count every time we find such a flow from $s1$ to $s2$. Therefore, the max flow count denotes the maximum number of edges going from $s1$ to $s2$, and this maximum number also suggests that the peg count is at its minimum which is $n - |Max|$. As long as the value of $k \geq n - |Max|$, it is guaranteed that we can put n discs onto k pegs. Therefore, the algorithm works correctly.

Running Time analysis:

General description: The running time is mostly contributed by the construction of the bipartite graph and finding the maximum matching.

1) Construction of the graph: $O(n) + O(n) + O(n^2) = O(n^2)$
2) Finding max match: $O(|V||E|) = O(n^3)$
3) Total Running Time : $O(n^3)$