1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE] This is the definition of NP.
Every problem in NP has a polynomial time certifier.

[FALSE] Decision problems can be in P.
Every decision problem is in NP-complete.

[TRUE] If 3-SAT reduces to it, it's NP-Hard and we now it's NP so it's NPC.
An NP problem is NP-complete if 3-SAT reduces to it in polynomial time.

[FALSE] It will be O(m^2 + mn) and not linear.
If all edges in a graph have capacity 1, then Ford-Fulkerson runs in linear time.

[FALSE] consider a graph with edges (a,b) = 4, (b,c) = 3 and (a, c) = 5. The shortest path from a to c is 5 but (a, c) is not in the MST.
Let T be a minimum spanning tree of G. Then, for any pair of vertices s and t, the shortest s-t path in G is the path in T.

[TRUE] Master's Theorem
If the running time of a divide-and-conquer algorithm satisfies the recurrence $T(n) = 3\,T(n/2) + \Theta(n^2)$, then $T(n) = \Theta(n^2)$.

[FALSE] Consider the problem in the previous item.
In a divide and conquer solution, the sub-problems are disjoint and are of the same size.

[FALSE] Many of them are NP-Complete.
All Integer linear programming problems can be solved in polynomial time.

[TRUE]
If the linear program is feasible and bounded, then there exists an optimal solution.

[FALSE] Each specific operation can be larger than O(log n).
Suppose we have a data structure where the amortized running time of Insert and Delete is O(lg n). Then in any sequence of 2n calls to Insert and Delete, the worst-case running time for the nth call is O(lg n).