Assignment #7

1. 34.1-6: do union, intersection, complementation, and concatenation.

Solution:
Complementation: Must show that \( \{ L \in P \} = \{ \overline{L} \in P \} \)

\( L \in P \) implies that there is an algorithm \( A \) that decides \( L \) and the time taken by \( A \) is given by \( T(n) = O(n^k) \). This implies that there is a constant \( c \) such that \( T(n) \leq cn^k \). Now given a string \( x \in \{0,1\}^* \) run algorithm \( A \) till either it accepts or rejects \( x \). If it accepts then output \( x \notin L \) else output \( x \in L \). Thus, there is an algorithm \( A' \) that decides \( \overline{L} \) in polynomial time.

Rest is left to you.

2. 34.2-9

Solution:
To do this we must show \( \{ L \in P \} = \{ \overline{L} \in NP \} \).
But \( \{ L \in P \} = \{ \overline{L} \in P \} \) by 34.1-6. Also \( P \subseteq NP \). So the result follows.

3. 34.5-2

To show that this problem is in NP, we use the vector \( x \) as the certificate. We now have to check that it satisfies each constraint in the system \( Ax \leq b \) and this can be done in polynomial time. If no such vector exists, we can not do this and hence there is no false certificate possible. To show that it is NP-complete, we show that 3-SAT \( \leq_p \) 0 − 1IP. To do this for each variable in 3-SAT we have a variable in 0 − 1IP and for each clause we have a constraint. For example, if a clause looks like \( (x_3 \lor \neg x_5 \lor x_8) \) the corresponding constraint would be \( y_3 + (1 - y_5) + y_8 \geq 1 \). The size of this IP is polynomially related to the boolean expression and the instance of 3-SAT has a yes answer iff each clause evaluates to 1. This happens iff the IP has a solution. The reduction is complete.

4. 34.5-5

Solution:
Certificate is \( A \) itself. Given \( A \) it is easy to check if \( A \) is the required set in polynomial time by adding numbers in \( A \) and \( S - A \). [Please note that this addition has an algorithm whose complexity is polynomial in the number of bits needed to represent all these numbers.] This is important since the problem is encoded in binary!] Hence SET-PARTITION \( \in \) NP.

To show that SET-PARTITION \( \in \) NPC: will show SUBSET-SUM \( \leq_p \) SET-PARTITION.

Let the instance of SUBSET-SUM be \( (S,t) \). In this problem we want to know if \( \exists B \subseteq S : \sum_{x \in B} x = t \).
Let \( \tilde{S} = S \cup \{t+1\} \cup \{(\sum_{x \in S} x) - t + 1\} \) and consider the SET-PARTITION problem with \( \tilde{S} \).

If the answer to this partition problem is "yes" with \( A \subseteq \tilde{S} \), then \( \sum_{y \in A} y = \sum_{y \in \tilde{S} - A} y \). Exactly one of the last two elements in \( \tilde{S} \) is in \( A \) and the other is in \( \tilde{S} - A \). This is because their sum exceeds the sum of all other elements (and all elements are positive). For the sake of specificity, let \( \{t+1\} \in A \). This implies \( \{(\sum_{x \in S} x) - t + 1\} \in \tilde{S} - A \). Hence,

\[
(\sum_{y \in A} y) + (t + 1) = (\sum_{y \in S - A} y) + (\sum_{y \in S} y) - t + 1
\]

\[
\implies (\sum_{y \in S - A} y) = t
\]

If the answer to SUBSET-SUM is yes, let \( B \subseteq S \) satisfy the relation that \( \sum_{y \in B} y = t \). Let \( \tilde{S} - A \) in SET-PARTITION be \( B \cup \{(\sum_{x \in S} x) - t + 1\} \) it is easy to show that this works. Thus, \( \text{SUBSET-SUM} \leq_p \text{SET-PARTITION} \).

can be computed in polynomial time.

5. 34-1: (a),(b)

(a) The corresponding decision problem is the following: Given \( G = [V, E] \), and a positive integer \( k \), is there an independence set in \( G \) of size \( k \) is the decision problem. Thus

\[
\text{INDEPENDENCE} = \{(G, k) : G \text{ is a graph with an independence set of size } k \}
\]

To show that it is in NP, we use the independence set itself as the certificate. It is easy to verify that a given set is indeed an independent set or not and check its size in polynomial time.

To show that it is NP-complete we show that \( \text{CLIQUE} \leq_p \text{INDEPENDENCE} \).

Given an instance of \( \text{CLIQUE} \) with \( \langle G, k \rangle \), let \( \overline{G} \) be the complement graph of \( G \). Let the corresponding \( \text{INDEPENDENCE} \) instance be on the \( \langle \overline{G}, k \rangle \). There is a clique of size \( k \) in graph \( G \) iff there is an independent set of size \( k \) in \( \overline{G} \). In fact the same set does it. Hence the reduction is complete.

(b) Run \( \text{INDEPENDENCE} \) over \( G \) with values of \( k \) in a binary search manner.