Problem 1

An NP complete problem belongs to both NP and NP-Hard (NPH). Thus, we must show that TSP belongs to both NP and NPH.

The TPS problem uses a complete graph (G = (V,E)) to find if there is a Hamiltonian cycle (C) that has the sum of weights along the edges less than or equal to a positive integer k.

Proving that TSP is an NP problem *(Its solution can be verified in polynomial time)*

1. Proving that TSP is a Hamiltonian cycle (C)
    - Proving that C is a complete graph can be done by using BFS to create a spanning tree of C (this runs in O (m' + n') where m' and n' are the edges and vertices of C) to create a spanning tree T with m_t edges and then comparing if m' > m_t to determine whether it's a cycle.
    - Proving that C contains all vertices of G runs in O (n) time.
    - Proving that all edges of C are also edges in G runs in O (m) time.

This can all be done in linear time.

2. Proving that the sum of the weights is less than or equal to k runs in O (m) time in the worst case. Since verifying the solution of TSP runs in polynomial time, then TSP is an NP problem.

Proving that TSP is an NPH problem

Assuming R is an NP problem, we must show that R --poly TSP. It is true that R --poly HC since HC is an NP complete problem. Also, HC--poly TSP. Thus, R --poly HC --poly TSP.

So, R --poly TSP.

Since TSP has passed both the conditions of being NP and NPH, then TSP is an NP-Complete problem.

Problem 2

Let SS be a subset problem of size n consisting of the set $S = \{S_0, S_1, S_2 \, --- \, S_n\}$ of positive Integers and a non negative Integer k.

The task is to transform S into a Knapsack problem (KN) in Polynomial time.

Let set S from SS be the KN set of items. for each i, let $w_i = s_i$ and let $V_i = s_i$. let $W = k$ and let $V = k$ where W is the maximum weight and V is the minimum Value. W is an Instance of $O(n)$.

These definitions specify KN as an Instance of $O(n)$.

<u>Proving solutions</u>

Solution that $SS \Rightarrow KN$

suppose T is a solution to SS (T is a subset of S whose sum is k)

$$\sum_{s_i \in T} s_i = k$$

Also

$$\sum_{s_i \in T} w_i = \sum_{s_i \in T} s_i = k = W$$

and

$$\sum_{s_i \in T} v_i = \sum_{s_i \in T} s_i = k = V$$

Thus the solution to SS yields the solution to KN

Verifying that solution to KN $\Rightarrow$ SS

Suppose T is a solution to KN

$$\sum_{s_i \in T} s_i = \sum_{s_i \in T} w_i \leq W = k$$

$$\sum_{s_i \in T} s_i = \sum_{s_i \in T} v_i \geq V = k$$

Thus $$\sum_{s_i \in T} s_i = k.$$

Thus

$SS \xrightarrow{\text{Poly}} KN$

Problem 3

Minimum vertex cover s = {B, D}

Vertex cover approximation = {A, B, C, D} = 2*s

Problem 4

Vertex cover for a graph G=(V, E) is a set C which is a subset of V, such that every edge e which is an element of E has at least one end vertex in C. This can be obtained by getting the power set P of V and checking for the subset with the minimum number of vertices that satisfies the vertex cover definition. This algorithm runs in O (2^n).

Given a problem with input G and k where is a non-negative integer, to verify the solution of vertex cover, the following steps will be taken

1. Check that all vertices in C are elements of V. This runs in O (n) time.
2. Check that every edge e of E has at least one vertex in C. This takes O (m * n). m = $O(n^2)$ for dense graph. Thus, this step takes $O(n^2)$ time.
3. Checking if the number of vertices is less than k, takes O (1) time.

Therefore, the verification process takes $O(n^2)$ which is polynomial time. This means that the vertex cover problem is an NP problem.