Only of myself I know how to tell,  
my world is as narrow as an ant’s.  
like an ant too my burden I carry,  
too great and heavy for my frail shoulder.

My way too - like the ant’s to the treetop -  
is a way of pain and toil;  
a gigantic hand, assured and malicious,  
a mocking hand hinders

All my paths are made bleak and tearful  
by the constant dread of this giant hand.

Why do you call to me, wondrous shores?  
Why do you lie to me, distant lights?  
– Only of Myself, Rachel

Required Problems

1. Poly time subroutines can lead to exponential algorithms. (20 pts.)
Show that an algorithm that makes at most a constant number of calls to polynomial-time subroutines runs in polynomial time, but that a polynomial number of calls to polynomial-time subroutines may result in an exponential-time algorithm.

2. Beware of algorithms carrying oracles. (50 pts.)
Consider the following optimization problems, and for each one of them do the following:
   (I) (2 pts.) State the natural decision problem corresponding to this optimization problem.
   (II) (3 pts.) Prove that this decision problem is \textsc{NP-Complete} by showing a reduction from one of the \textsc{NP-Complete} problems seen in class. If you already seen this problem in class state “seen in class” and move on with your life.
Assume that you are given an algorithm that can solve the decision problem in polynomial time. Show how to solve the original optimization problem using this algorithm in polynomial time.

Prove that the following problems are **NP-Complete**.

(A) (10 pts.)

**SET COVER**

*Instance*: Collection \( C \) of subsets of a finite set \( S \).

*Target*: Compute the minimum \( k \), and the sets \( S_1, \ldots, S_k \) in \( C \), such that \( S \subseteq \bigcup_{i=1}^{k} S_i \).

(B) (10 pts.)

**MIN BIN PACKING**

*Instance*: Finite set \( U \) of items, a size \( s(u) \in \mathbb{Z}^+ \) for each \( u \in U \), an integer bin capacity \( B \)

*Target*: Compute the minimum \( k \), and a partition of \( U \) into disjoint sets \( U_1, \ldots, U_k \), such that the sum of the sizes of the items inside each \( U_i \) is \( B \) or less.

(C) (10 pts.)

**HITTING SET**

*Instance*: A *ground set* \( U = \{1, \ldots, n\} \), and a set \( \mathcal{F} = \{U_1, \ldots, U_m\} \) of subsets of \( U \).

*Target*: Find the smallest set \( S' \subseteq U \), such that \( S' \) hits all the sets of \( \mathcal{F} \). Specifically, \( S' \subseteq U \) is a *hitting set* if for all \( U_i \in \mathcal{F} \), we have that \( S' \) contains at least one element of \( U_i \).

(D) (10 pts.)

**Max Degree Spanning Tree**

*Instance*: Graph \( G = (V, E) \).

*Target*: Compute the spanning tree \( T \) in \( G \) where the maximum degree of a vertex in \( T \) is minimized.

(E) (10 pts.)

**Partition into independent sets.**

*Instance*: Graph \( G = (V, E) \).

*Target*: Compute the minimum \( k \), and the partition of \( V \) into \( k \) sets \( V_1, \ldots, V_k \), such that for every edge \( uv \) of \( G \), there exists two distinct indices \( i \) and \( j \), such that \( u \in V_i \), and \( v \in V_j \).

### 3. Dominating set. (30 pts.)

Consider the following problem (which is **NP-Complete**):

**Dominating set**

*Instance*: Graph \( G = (V, E) \) and an integer \( k \).

*Target*: Is there a subset \( X \subseteq V \) of size \( k \), such that each vertex of \( V \setminus X \) is adjacent to some vertex of \( X \).

Consider the variant where \( V = \{1, \ldots, n\} \), and \( ij \in E \), implies that \(|i - j| \leq 8\). We refer to this variant as the 8Dominating set problem.

Either prove that 8Dominating set is **NP-Complete**, or alternatively, provide a polynomial time algorithm for solving it.