Below is a set of practice problems on NP-completeness and reduction theory. These exercises refer to several well-known NP-complete problems, most of which we will talk about in class: SAT, 3-CNF-SAT, SUBSET-SUM, TSP, VERTEX-COVER, and CLIQUE.

1. Two well-known NP-complete problems are 3-CNF-SAT and TSP, the traveling salesman problem. The 2-CNF-SAT problem is a SAT variant in which each clause contains at most 2 literals; it is known to have a polynomial-time algorithm. Is each of the following statements true or false? Justify your answer.

   (a) \(3\text{-CNF-SAT} \leq_p \text{TSP}\).

   (b) If \(P \neq NP\), then \(3\text{-CNF-SAT} \leq_p 2\text{-CNF-SAT}\).

   (c) If \(P \neq NP\), then no NP-complete problem can be solved in polynomial time.

2. Prove that the following problem, the Non-Bored Jogger Problem (NBJ), is NP-complete. You are given as input a weighted, undirected multigraph \(G\); a distinguished home node \(v\) in \(G\), and an integer \(\ell \geq 0\). Each edge in \(G\) has a positive integer weight. Self-loops are permitted in \(G\), as are multiple edges with the same endpoints. Does there exist a path in \(G\) that starts and ends at \(v\), traverses a set of edges with total weight \(\ell\), and traverses each edge at most once? (Hint: Reduce from SUBSET-SUM.)

3. The set intersection problem (SIP) is defined as follows: Given finite sets \(A_1, A_2, \ldots, A_r\) and \(B_1, B_2, \ldots, B_s\), is there a set \(T\) such that

   \[|T \cap A_i| \geq 1 \text{ for } i = 1, 2, \ldots, r\]

   and

   \[|T \cap B_j| \leq 1 \text{ for } j = 1, 2, \ldots, s?\]

   Prove that the set intersection problem is NP-complete. (Hint: Reduce from 3-CNF-SAT).

4. Consider the following “gold-digger problem” (GDP). You are given a map of a territory consisting of a set of towns connected by trails. Each trail \((u, v)\) connecting towns \(u\) and \(v\) is labeled with a dollar value \(w(u, v)\), which is the value of the gold you will find along that trail. You can traverse a trail as often as you want, but you only get the value of the trail the first time you traverse it; subsequent traversals have no value. Each town \(v\) has a lodging cost \(c(v)\), which you pay each time you enter the town.

   An expedition is a cyclic path that starts and ends at a given town. An expedition has profit \(k\) if the total value of gold found, minus the cost of the all the town visits, is \(k\). The goal is to find an expedition of maximum profit.

   Either show that there exists a polynomial-time algorithm for GDP, or show that the corresponding decision problem is NP-complete. If you want to show that the problem is hard, you
may reduce from any of: SAT, 3-CNF-SAT, CLIQUE, VERTEX-COVER, SUBSET-SUM, PARTITION, HAM-CYCLE, TSP.