Exercise 12.15. It's-a me, Mario! This problem is inspired by Super Mario World, where Mario must make it from some starting point to the flag in front of a castle, collecting as many coins as possible along the way.

Let $G = (V, E)$ be a directed graph where each vertex $v \in V$ has $C_v \geq 0$ coins. For a walk in G, we say that the *number of coins collected by the walk* is the total sum of C_v over all *distinct* vertices v in the walk. (If we visit a vertex v more than once, we still only get C_v coins total.) Given $s, t \in V$, the goal is to compute the maximum number of coins collected by any (s, t)-walk.

Exercise 12.15.1. Let G be a DAG. For this problem, either (a) design and analyze a polynomial time algorithm (the faster the better), or (b) prove that a polynomial time algorithm would imply a polynomial time algorithm for SAT.

Solution. solution □

Exercise 12.15.2. Let G be a DAG. For this problem, either (a) design and analyze a polynomial time algorithm (the faster the better), or (b) prove that a polynomial time algorithm would imply a polynomial time algorithm for SAT.

Solution. Let G be a general directed graph. For this problem, either (a) design and analyze a polynomial time algorithm (the faster the better), or (b) prove that a polynomial time algorithm would imply a polynomial time algorithm for SAT. □

Exercise 13.4. Let $G = (V, E)$ be a directed graph. We say that a set of vertices is *almost independent* if each $v \in S$ has at most one neighbor in S.[5] Consider the problem of computing the maximum cardinality of any almost independent set of vertices. For this problem, either (a) design and analyze a polynomial time algorithm (the faster the better), or (b) prove that a polynomial time algorithm would imply a polynomial time algorithm for SAT.

[5]Two vertices u and v are neighbors if they are connected by an edge.

Solution. solution □

Exercise 13.6. Recall the dominating set problem from section 13.3[†]. Here we will consider the weighted version where the vertices are given positive weights, and the goal is to compute the minimum weight dominating set. For each of the following problems, either (a) design and analyze a polynomial time algorithm (the faster the better), or (b) prove that a polynomial time algorithm would imply a polynomial time algorithm for SAT.

[†]A set of vertices $S \subseteq V$ is a *dominating set* if every vertex $v \in V$ is either in S or the neighbor of a vertex in S. The minimum dominating set problem is to compute the minimum cardinality dominating set.

Exercise 13.6.1. The minimum weight dominating set problem for intervals, with the additional assumption that no two intervals are nested. To state it more precisely: the input consists of n weighted intervals $\mathcal{I}$. The non-nested assumptions means that for any two intervals $I, J \in \mathcal{I}$, we never have I contained in J or J contained in I.

The goal is to compute the minimum weight subset $S \subseteq \mathcal{I}$ of intervals such that every interval in $\mathbb{I}$ is either in S or overlaps some interval in S.

- (For 1 pt. extra credit) Extend your algorithm to general intervals.[8]

[8]Of course, anyone who has already solved the general case automatically solves the special case where no two intervals are nested.

Solution. solution □

Exercise 13.6.2. The minimum weight dominating set problem in trees.

Solution. □