1. (Easy.)
   a Define the class \( P \).
   b Define the class \( NP \).

2. (Easy-Medium.) In the Set Cover problem, you are given a set \( U \) of ground elements \( e_1, e_2, \ldots, e_n \). You are also given a set \( S \) consisting of \( m \) subsets \( S_1, S_2, \ldots, S_m \) of \( U \). Given a positive integer \( k \), is it possible to select at most \( k \) subsets from \( S \) so that every element \( e_i \) from \( i = 1, \ldots, n \) is contained in at least one selected subset? I.e., is there a set of at most \( k \) subsets whose union covers \( U \)? Prove that Set Cover is NP-Complete. You may want to consider giving a reduction from Vertex Cover.

3. (Easy-Medium, from Quiz 2.) Suppose you want to hike from village \( V_1 \) to village \( V_N \) in \( K \) days. There are a number of intermediate villages along the way where you can stay overnight. Let \( d(i, i+1) \) denote the distance between villages \( V_i \) and \( V_{i+1} \). The goal is to minimize the maximum amount of walking you have to do in a single day.

   For example, consider the situation in which you need to go from \( V_1 \) to \( V_7 \) in 3 days (see figure). If you stop at \( V_3 \) and \( V_5 \), then on day 1, you walk from \( V_1 \) to \( V_3 \) (10 miles), on day 2 you walk from \( V_3 \) to \( V_5 \) (12 miles) and on day 3, you walk from \( V_5 \) to \( V_7 \) (9 miles). Then the maximum distance walked per day is 12 miles. A better solution is to walk from \( V_1 \) to \( V_4 \) on the first day (11 miles), from \( V_4 \) to \( V_5 \) on day 2 (11 miles) and \( V_5 \) to \( V_7 \) on day 3 (9 miles): the maximum distance walked per day is only 11 miles.

   Design a dynamic programming algorithm to minimize the maximum amount of walking in a single day. No proof required, but give time complexity and explain your recurrence.

4. (Medium-Hard.) Say you have access to a function \( \text{DICT} \) that returns true if its input is a valid English word, and false otherwise. We are given as input a sentence from which the punctuation has been stripped (for example, “dynamicprogrammingisfabulous”). Assuming calls to \( \text{DICT} \) take constant time, give an \( O(n^2) \) time algorithm to determine whether an input string of length \( n \) can be split into a sequence of valid words.
5. (Medium.) In the famous Traveling Salesman Problem (TSP), a salesman must visit \( n \) cities labeled \( v_1, v_2, \ldots, v_n \). The salesman starts in city \( v_1 \), his home, and wants to find a tour, i.e., an order in which to visit all the other cities and return home. Formally, given a set of distances of \( n \) cities, and a bound \( D \), is there a tour of length at most \( D \)? Distances are non-negative and need not be symmetric. Prove that TSP is NP-complete.

6. (Hard.) (Adapted from KT, p. 412) Suppose you have \( n \) doctors at a hospital who need to collectively cover the vacation days over the next year. There are \( k \) vacation periods, each spanning several contiguous days. Let \( D_j \) be the set of days included in the \( j \)th vacation period. Each doctor \( i \) has a set \( S_i \) of days when he or she can work; these availabilities need not form a contiguous time period, even within a single vacation period. For a fixed \( c \), each doctor should be assigned to work at most \( c \) vacation days total, and only on days when he or she is available. Also, for each vacation period \( j \), each doctor should be assigned to work at most one of the days in the set \( D_j \). Give a polynomial-time algorithm that determines whether it is possible to validly assign a single doctor to each vacation day and still cover every vacation day.