1. (20) Let $G = (V, E, w)$ be a connected weighted graph. Suppose that Dijkstra's algorithm is used to solve the single source shortest path problem on the weighted graph G.
   (a) Give an example to show that Dijkstra's algorithm does not work for graphs with negative weights.
   (b) Can we first add a constant value to the weight of each edge to make all weights non-negative, then run the Dijkstra's algorithm to compute shortest paths? Justify your answer.

2. (30) Let $G = (V, E, w)$ be a connected weighted undirected graph with positive weight on the edge set E. BellmanFord is based on dynamic programming approach. In its basic structure it is similar to Dijkstra's Algorithm, but instead of greedily selecting the minimum-weight node not yet processed to relax, it simply relaxes all the edges, and does this $n1$ times, where n is the number of vertices in the graph.
   (a) (10) Describe the Bellman-Ford's algorithm for single source shortest path problem on the graph G by using a C-like code.
   (b) (10) Briefly show that the algorithm correctly computes shortest paths from the start vertex s to other vertexes when there are no negative cycles in the graph.
   (c) (10) Show that the algorithm can correctly detect the existence of negative cycles in the graph.

3. (40) Given a string of characters $s_1 s_2 \ldots s_n$. It is believed that the string is a document in which all space and punctuation have been removed. Reconstruct the document using a dictionary, which is available in the form of Boolean function $d(w) = 1$ if and only if w is a word. Use the example

   `wewillmeetatmidnight`

   to show how this problem can be solved by the following methods.
   (a) Constructed a graph from the string $s_1 s_2 \ldots s_n$ and the find a path in the graph.
   (b) Solve the problem by dynamic programming.

   You can assume that the only words are

   `a at me meet mid midnight night we will`

4. (20) Let n be a product of two large primes p and q. Design a polynomial time nondeterministic algorithm for finding a factor of n, that is, p or q.

5. (20) The $\{0, 1\}$-*knapsack problem* is defined as:

   Given a set of $K = \{(w_i, v_i)\}$, $1 \le i \le n$, and a capacity of the knapsack S, find an n-dimensional vector $U = (u_1, u_2, \ldots, u_n)$ in $\{0, 1\}^n$ such that $W = \sum_{i=1}^{n} u_i \cdot w_i \le S$ and the value of $P = \sum_{i=1}^{n} u_i \cdot v_i$ is maximized.

   The *subset sum problem* is defined as:

   Given a sequence of n integers $X = x_1, x_2, \ldots, x_n$, and an integer S. Fine a subsequence X' of X such that the sum of the integers in X' is S, or report that there are no such subsequences.

   Show that the subset sum problem can be reduced to the $\{0, 1\}$-knapsack problem.

6. (20) Define problems $\mathcal{A}$ and $\mathcal{B}$ as follows.

   $\mathcal{A}$: **input**: two integers a and b. **output**: the product $a \times b$.
   $\mathcal{B}$: **input**: an integer a. **output**: the square a^2.

   (a) Show that problem $\mathcal{A}$ can be reduced to problem $\mathcal{B}$ in linear time.
   (b) Show that problem $\mathcal{B}$ can be reduced to problem $\mathcal{A}$ in linear time.
   (c) Is it possible to design an algorithm for computing a^2 which is asymptotically faster then computing $a \times b$?
   (d) Is it possible to design an algorithm for computing $a \times b$ which is asymptotically faster then computing a^2?