1. The Ford-Bellman algorithm. Consider a directed graph $G = (V, A)$, $V = \{1, 2, \ldots, n\}$ with given lengths of arcs c_{ij}. Recall that the shortest path from any vertex i to a fixed destination $n \in V$ can be detected once we know the table of shortest distances d_i from i to n. This can be done using (already discussed) Dijkstra's algorithm; however, it works if $c_{ij} \geqslant 0$ for every $(i, j) \in A$.

Sometimes we want to consider also negative lengths (imagine that c_{ij} is a price for the bus ticket from i to j and you have been promised to get double reimbursement for some travels). This can be done using the Ford-Bellman algorithm based on the following.

Let $d_i(t)$ denotes the length of the shortest path from i to n using at most t arcs. Note that $d_n(t) = 0$ for every t (we are already there); we put $d_i(t) = \infty$ if there are no path joining i and n by at most t arcs (in particular, $d_i(0) = \infty$ for $i \neq n$).

The main point is that

$$d_i(t+1) = \min_{k \in \mathrm{Out}(i)} (c_{ik} + d_k(t));$$

why? Check that this enables one to calculate the minimal distance d_i which is equal to $d_i(n-1)$. Note that this works even if some 'distances' are negative; we, however, need to assume that there are no cycles in the graph consisting of arcs of negative length.

2. The Floyd–Warshall algorithm. Consider the problem as above. Let $d_{ij}(k)$ denotes the shortest distance from i to j that goes only through vertices $\{1, 2, \ldots, k\}$ (as intermmediate points of the travel). Prove that

$$d_{ij}(k+1) = \min\left(d_{ij}(k),\ d_{i,k+1}(k) + d_{k+1,j}(k)\right).$$

How to design an algorithm for finding d_{ij}?

3. Shannon switching game. The game is played on a connected (undirected) graph in which there is vertex a (=prison) an another one b (=escape). There are two players: Sheriff and Prisoner. Sheriff starts and marks one edge from the graph, say by $-$. Then Prisoner marks one of remaining edges, say by $+$. Then Sheriff marks one of the remaining edges and so on.

At the end all the edges are marked either $+$ or $-$; Prisoner wins if he can escape, that is there is a path from a to b using $+$ edges. Otherwise, Sheriff wins.

Assume that in a graph contains two spanning trees $T(0)$ and $S(0)$ having disjoint sets of edges. Show that Prisoner has a winning strategy: when one edge gets $-$ then he can replace $T(0)$ and $S(0)$ by two spanning trees having one common edge which is marked $+$. Continuing in this fashion, at the end he gets a spanning tree marked $+$ so there is an escape.

Above we use the fact that if we incorporate a new edge to some spanning tree then we get a cycle and we can form another spanning tree by removing some old edge. It is a good idea to play first on some small example — one may try to google online implementations of the game.