1. In some country there are n cities and m bidirectional roads between them. Each city has an army. Army of the i-th city consists of a_i soldiers. Now soldiers roam. After roaming each soldier has to either stay in his city or to go to the one of neighboring cities by moving along at most one road. Check if it is possible that after roaming there will be exactly b_i soldiers in the i-th city.

   [Solution sketch] Model it as a maximum flow problem as follows. Let's build a flow network in following way. Make a source. Make a first group of vertices consisting of n vertices, each of them for one city. Connect a source with i-th vertex in first group with edge that has capacity a_i**.** Make a sink and a second group of vertices in the same way, but use b_i except for a_i for connecting the vertices with the sink. If there is a road between cities i and j or i = j, make two edges, first should be connecting the i-th vertex from first group to the j-th vertex from second group, and has infinity capacity. Second should be similar, but connect j-th vertex from first group to i-th vertex from second group. Then find a maxflow in this graph. If maxflow is equal to sum of a_i (which is equal to sum of b_i), then the answer is yes, otherwise no.

2. Prove that every k-regular bipartite graph has a perfect matching (prove using notions of max flow).

   [Solution sketch] Let X and Y be the two partitions. First note that the graph is k-regular implies that |X| = |Y| (easy to prove by counting edges going from each side to the other, which must be the same for both sides). Create a flow graph same as you do for finding maximum matching in a bipartite graph using max flow. Then assign flows to each edge as follows. For all edges (s,u), u in X and (v, t), v in Y, assign flow =1. For all edges (u,v), u in X and v in Y, assign flow = 1/k. Verify that it is a feasible flow. Also, it is a max-flow because all edges from s (or into t) are saturated. Value of this flow = |X| = |Y|, so this indicates a perfect matching (max-flow value = max-matching size). This proves that the perfect matching exists. If you want to find the matching, use the algorithm.

3. A weighted cover of a weighted bipartite graph is an assignment of costs x(i) to each vertex i such that for any edge (i,j), x(i) + x(j) ≥ w(i,j), the weight of the edge (i,j). The cost of the cover is the sum of the costs of all the vertices. Find a weighted cover of a bipartite graph with minimum cost.

   [Solution sketch] Note that a weighted cover is nothing but the sum of the labels we have seen in the Hungarian algorithm, with the w's same as the labels. Apply Hungarian algorithm to find the maximum weighted matching (adding dummy edges etc. to make the graph complete as usual if not already so). The labels on the nodes at the end of the Hungarian algorithm is the required minimum weighted cover. This is because sum of the weights of the edges in the maximum weighted (perfect) matching found is equal to the sum of the labels of the endpoints of the edges in the matching (proved in the study material given), so the min weighted cover cannot be larger (as then the maximum weighted matching is not maximum). It is easy to see it cannot be smaller, as then it is not a cover.

4. For the stable matching problem, show an example in which more than one stable matching exists.

   [Solution]: Consider the sets {A, B, C} and {X, Y, Z} with the preference lists as below.

   - X: A, B, C   Y: B, A, C   Z: A, B, C
   - A: Y, X, Z   B: X, Y, Z   C: X, Y, Z

   Then both of the following are valid stable matchings:

   - A-X, B-Y, C-Z
   - A-Y, B-X, C-Z