Question 1

(a) The value of this flow is 10. It is not a maximum flow.

(b) The minimum cut is $(\{s, a, b, c\}, \{d, t\})$. Its capacity is 11.

Question 2

This is false. Consider a graph with nodes s, v_1, v_2, v_3, w, t, edges (s, v_i) and (v_i, w) for each i, and an edge (w, t). There is a capacity of 4 on edge (w, t), and a capacity of 1 on all other edges. Then setting $A = \{s\}$ and $B = V - A$ gives a minimum cut, with capacity 3. But if we add one to every edge then this cut has capacity 6, more than the capacity of 5 on the cut with $B = \{t\}$ and $A = V - B$.

Question 3

We build the following flow network. There is a node v_i for each client i, a node w_j for each base station j, and an edge (v_i, w_j) of capacity 1 if client i is within range of base station j. We then connect a super-source s to each of the client nodes by an edge of capacity 1, and we connect each of the base station nodes to a super-sink t by an edge of capacity L.

We claim that there is a feasible way to connect all clients to base stations if and only if there is an s-t flow of value n. If there is a feasible connection, then we send one unit of flow from s to t along each of the paths s, v_i, w_j, t, where client i is connected to base station j. This does not violate the capacity conditions, in particular on the edges (w_j, t), due to the load constraints. Conversely, if there is a flow of value n, then there is one with integer values. We connect client i to base station j if the edge (v_i, w_j) carries one unit of flow, and we observe that the capacity condition ensures that no base station is overloaded.

The running is the time required to solve a max-flow problem on a graph with $O(n + k)$ nodes and $O(nk)$ edges.

Question 4

For each node v other than the source and sink, we replace it with two nodes, v_{in} and v_{out}. All edges that used to come into v now go into v_{in}, and all edges that used to come out of v now go out of v_{out}. All these edges have infinite capacity. (Or if is enough to choose a number larger than the sum of all node capacities.) Finally, there is an edge (v_{in}, v_{out}) of capacity c_v. We consider flows from s_{in} to t_{out} in this new graph.

Now, if there is a flow of value ν in this new graph, then there is a flow of value ν in the original graph that respects all node capacities: we simply use the flow obtained by contracting all edges of the form (v_{in}, v_{out}). Conversely, if there is a flow of value ν in the original graph, then we can use it to construct a flow of value ν in this new graph; the flow using v in the original graph will now pass through the edge (v_{in}, v_{out}), and it will not exceed this edge capacity due to the node capacity condition in the original graph.

Thus, to compute a maximum flow subject to the node capacities, we simply compute a standard maximum flow in the new graph.

Now, for the version of the Max-Flow Min-Cut Theorem for node-capacitated graphs. If we apply the standard Max-Flow Min-Cut Theorem to the new graph we constructed, it says that the maximum flow equals the minimum capacity of a cut. A minimum cut will only cut edges of the form (v_{in}, v_{out}) — cutting all such edges will separate s_{in} from t_{out}, and cutting any single other edge will produce a more expensive cut. Interpreted in the original graph, this corresponds to deleting a subset of the nodes, and defining the capacity of this deleted set to be the sum of the capacities of the deleted nodes.

This shows that in a node-capacitated network, the value of the maximum flow is equal to the minimum capacity of a set of nodes whose deletion leaves no path from s to t.