(a) The flow has value 18. It is not a maximum flow.

(b) Define the set A to be s and the top node. The cut $(A, V - A)$ is a minimum cut and has capacity 21.

[1] ex700.306.334

(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.

[1] ex84.475.557

This is is false. Consider a graph with nodes s, v, w, t, edges (s, v), (v, w), (w, t), capacities of 2 on (s, v) and (w, t), and a capacity of 1 on (v, w). Then the maximum flow has value 1, and does not saturate the edge out of s.

[1] ex948.608.156

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 $A = V - B$.

[1]ex820.292.535

We build the following bipartite graph $G = (V, E)$. V is partitioned into sets X and Y, with a node $x_i \in X$ representing switch i, and a node $y_j \in Y$ representing fixture j. (x_i, y_j) is an edge in E if and only if the line segment from x_i to y_j does not intersect any of the m walls in the floor plan. Thus, whether $(x_i, y_j) \in E$ can be determined initially by m segment-intersection tests; so G can be built in time $O(n^2 m)$.

Now, we test in $O(n^3)$ time whether G has a perfect matching, and declare the floor plan to be "ergonomic" if and only if G does have a perfect matching. Our answer is always correct, since a perfect matching in G is a pairing of the n switches and the n fixtures in such a way that each switch can see the fixture it is paired with, by the definition of the edge set E; conversely, such a pairing of switches and fixtures defines a perfect matching in G.

[1]ex527.636.149

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.

[1]ex751.45.676

To solve this problem, construct a graph $G = (V, E)$ as follows: Let the set of vertices consist of a super-source node, four supply nodes (one for each blood type) adjacent to the source, four demand nodes and a super sink node that is adjacent to the demand nodes. For each supply node u and demand node v, construct an edge (u, v) if type v can receive blood from type u and set the capacity to ∞ or the demand for type v. Construct an edge (s, u) between the source s and each supply node u with the capacity set to the available supply of type u. Similarly, for each demand node v and the sink t, construct an edge (v, t) with the capacity set to the demand for type v.

Now compute an (integer-valued) maximum flow on this graph. Since the graph has constant size, the scaling max-flow algorithm takes time $O(\log C)$, where C is the total supply, and the Preflow-Push algorithm takes constant time.

We claim that there is sufficient supply for the projected need. if and only if the edges from the demand nodes to the sink are all saturated in the resulting max-flow. Indeed, if there is sufficient supply, in which s_{ST} of type S are used for type T, then we can send a flow of s_{ST} from the supply node of type S to the demand node of type T, and respect all capacity conditions. Conversely, if there is a flow saturating all edges from demand nodes to the sink, then there is an integer flow with this property; if it sends f_{ST} units of flow from the supply node for type S to the demand node for type T, then we can use f_{ST} nodes of type S for patients of type T.

(b) Consider a cut containing the source, and the supply and demand nodes for B and A. The capacity of this cut is $50 + 36 + 10 + 3 = 99$, and hence all 100 units of demand cannot be satisfied.

An explanation for the clinic administrators: There are 87 people with demand for blood types O and A; these can only be satisfied by donors with blood types O and A; and there are only 86 such donors.

Note. We observe that part (a) can also be solved by a greedy algorithm; basically, it works as follows. The O group can only receive blood from O donors; so if the O group is not satisfied, there is no solution. Otherwise, satisfy the A and B groups using the leftovers from the O group; if this is not possible, there is no solution. Finally, satisfy the AB group using any remaining leftovers. A short explanation of correctness (basically following the above reasoning) is necessary for this algorithm, as it was with the flow algorithm.

[1] ex717.885.42

We build the following flow network. There is a node v_i for each patient i, a node w_j for each hospital j, and an edge (v_i, w_j) of capacity 1 if patient i is within a half hour drive of hospital j. We then connect a super-source s to each of the patient nodes by an edge of capacity 1, and we connect each of the hospital nodes to a super-sink t by an edge of capacity $\lceil n/k \rceil$.

We claim that there is a feasible way to send all patients to hospitals if and only if there is an s-t flow of value n. If there is a feasible way to send patients, then we send one unit of flow from s to t along each of the paths s, v_i, w_j, t, where patient i is sent to hospital 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 send patient i to hospital j if the edge (v_i, w_j) carries one unit of flow, and we observe that the capacity condition ensures that no hospital 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.

[1]ex297.369.93

We will assume that the flow f is integer-valued. Let $e^* = (v, w)$. If the edge e^* is not saturated with flow, then reducing its capacity by one unit does not cause a problem. So assume it is saturated.

We first reduce the flow on e^* to satisfy the capacity conditions. We now have to restore the capacity conditions. We construct a path from w to t such that all edges carry flow, and we reduce the flow by one unit on each of these edges. We then do the same thing from v back to s. Note that because the flow f is acyclic, we do not encounter any edge twice in this process, so all edges we traverse have their flow reduced by exactly one unit, and the capacity condition is restored.

Let f' be the current flow. We have to decide whether f' is a maximum flow, or whether the flow value can be increased. Since f was a maximum flow, and the value of f' is only one unit less than f, we attempt to find a single augmenting path from s to t in the residual graph $G_{f'}$. If we fail to find one, then f' is maximum. Else, the flow is augmented to have a value at least that of f; since the current flow network cannot have a larger maximum flow value than the original one, this is a maximum flow.

[1] ex257.178.863

The statement is false and the following is a counterexample to it. Let us be given a number $b > 1$ (we will without loss of generality assume that it is an integer, otherwise we will round it up). We consider the following graph. It has $2(b+1)+2$ vertices: source s sink t, and vertices $u_1, u_2, \ldots, u_{b+1}$ that have an edge coming from the source and vertices $v_1, v_2, \ldots, v_{b+1}$ that have an edge going into the sink. There is also an edge from u_i to v_i and from v_i to u_{i+1}. All the edge capacities are 1.

Now assume that the first augmenting path was the path $s \to u_1 \to v_1 \to u_2 \to v_2 \to \ldots u_{b+1} \to v_{b+1} \to t$. Then since all the backward edges are deleted from the residual graph according to the super-fast algorithm, the residual graph would contain no path from s to t, and therefore our final flow would equal 1. But there is a flow of value $b+1$ by using the horizontal edges (that is $u_i \to v_i$). Therefore we failed to reach within b of the optimum.

Notice that for different b's we would be considering different graphs, but we are allowed to do this, since the problem asks whether there exists a *universal* b that is independent of the choice of the flow graph G.

[1]ex70.281.132

If the minimum s-t cut has size $\leq k$, then we can reduce the flow to 0. Otherwise, let $f > k$ be the value of the maximum s-t flow. We identify a minimum s-t cut (A, B), and delete k of the edges out of A. The resulting subgraph has a maximum flow value of at most $f - k$.

But we claim that for any set of edges F of size k, the subgraph $G' = (V, E - F)$ has an s-t flow of value at least $f - k$. Indeed, consider any cut (A, B) of G'. There are at least f edges out of A in G, and at most k have been deleted, so there are at least $f - k$ edges out of A in G'. Thus, the minimum cut in G' has value at least $f - k$, and so there is a flow of at least this value.

[1] ex225.750.725