Problem 1. Construct a network with source $s$, category node $C_1, \ldots, C_{10}$, question nodes $Q_1, \ldots, Q_{100}$ and a sink $t$. Connect $s$ to each $C_i$ with capacity 10 edges. Connect each $Q_j$ to $t$ with capacity 1 edges. Connect question $Q_j$ to $C_i$ if that question has those corresponding categories. One can make a question paper if of 100 questions if and only if there is the maximum flow of the network has value 100. Such a maximum flow can be found, for instance, by means of the Ford-Fulkerson algorithm.

Problem 2.

1. Split the vertex $v$ into two vertices $v_{in}$ and $v_{out}$ and join them with an edge of capacity equal to the node capacity.

2. If the capacity of an edge $e = (u, v)$ is $c_e$ and the lower bound is $l_e$, then define an equivalent network $N'$ on the same node and edge set such that the capacity of $e$ is $c'_e = c_e - l_e$ and lower bound is 0 (the standard flow problem). Further, we add an extra source $s'_e$ and an extra sink $t'_e$ for each edge, s.t. $u$ is connected to $t'_e$ by a link of capacity $l_e$ and $s'_e$ is connected to $v$ by a link of capacity $l_e$. This equivalent network has multiple sources and sinks and, therefore, it can be reduced to a new network $N''$ with a single source and a single sink, as seen in class.

Problem 3.

1. Suppose that there are $M$ people that need to be moved out. First, we provide an algorithm to decide if all people can be moved out in $T$ steps. Given this algorithm, we can do a binary search on $T$ between 1 to $|V|/M/c$ to find the shortest time in which all the people can move out. Our algorithm is as follows: given the graph $G$, we construct $G_T$ as follows. For each $v \in V$, we make $T$ copies of $v$: $v_1, \ldots, v_T$, where copy $v_i$ corresponds to time step $i$. For each $v$, we construct an edge from $v_i$ to $v_{i+1}$ with infinite capacity (people can just stay in rooms at a time step). We then construct an edge from $v_i$ to $w_{i+1}$ with capacity $c$ if there exists an edge from $v$ to $w$ with capacity $c$ in $G$. Suppose everyone is in room $a$ initially, and the exit is room $b$. Then we set the source $s = a_1$, and the sink $t = b_T$. To test if all the people can get from the source to the sink in $T$ time steps, we check if the max flow in $G_T$ is greater than or equal to the number of people initially at the source. If so, we can move all the people across this graph in $T$ time steps.

2. We can use the same overall idea: construct a graph $G_T$, and compute its max flow. The construction of $G_T$ is the same, except for the following. We create a source $s$ and sink $t$. Let $S$ be the start vertices corresponding to the rooms that initially contain all the people, and let $U$ be the sink vertices that correspond to all the exits. We create a link from $s$ to each $x_1$, for each $x \in S$ with capacity equal to the number of people starting at $x$. Similarly, we create a link from each $x_T$ (for each $x \in U$) to $t$ with infinite capacity.

3. Again, the overall idea is the same. But when we construct $G_T$ now, we create edges between the layers in a different way: construct the edge linking $v_i$ to $w_{i+1}(v, w)$ with capacity $c$ if there is an edge between $v$ and $w$ in $G$ with transit time $t(v, w)$.

Problem 4. Let us state the vertex version of Menger’s theorem: if $u$ and $v$ are non-adjacent vertices of
a graph, the maximum number of internally disjoint \((u, v)\)-paths in \(G\) is equal to the minimum number of vertices whose deletion destroys all \((u, v)\)-paths.

As \(G\) is \(k\)-connected and \(Y\) contains at least \(k\) vertices, the claim follows by applying the vertex version of Menger’s theorem to \(N(x)\) (the set of neighbors of \(x \in G\)) and \(Y\).