Consider the directed graph $G = (V, E)$ with non-negative capacities $c : E \to \mathbb{R}_{\geq 0}$ and an $(s, t)$-flow $f : E \to \mathbb{R}_{\geq 0}$ that is feasible with respect to $c$.

(a) Draw the residual graph $G_f = (V_f, E_f)$ for flow $f$. Be sure to label every edge of $G_f$ with its residual capacity.

Solution:

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{residual_graph.png}
\caption{The residual graph $G_f$.}
\end{figure}

Rubric: 2 points total.

(b) Describe an augmenting path $s = v_0 \to v_1 \to \ldots \to v_r = t$ in $G_f$ by either drawing the path in your residual graph or listing the path's vertices in order.

Solution: There is an augmenting path $s \to a \to c \to d \to t$.

Rubric: 2 points total.
(c) Let $F = \min_i c_f(v_i \rightarrow v_{i+1})$ and let $f' : E \rightarrow \mathbb{R}_{\geq 0}$ be the flow obtained from $f$ by pushing $F$ units through your augmenting path. Draw a new copy of $G$, and label its edges with the flow values for $f'$.

Solution:

![Graph](image)

Figure 2. The $(s, t)$-flow $f'$.

Rubric: 2 points total.

(d) Draw the residual graph $G_{f'} = (V, E_{f'})$ for flow $f'$.

Solution:

![Graph](image)

Figure 3. The residual graph $G_{f'}$. 
(e) There shouldn’t be any augmenting paths in $G_{f'}$, implying $f'$ is a maximum flow. Draw or list the vertices in $S$ for some minimum $(s, t)$-cut $(S, T)$.

**Solution:** $S = \{s, a, b, c\}$, the set of vertices reachable from $s$. 

(f) What is the value of the maximum flow/capacity of the minimum cut?

**Solution:** The value/capacity is 8.
Consider the following generalization of the bipartite matching problem. You are given an undirected bipartite graph $G = (L \cup R, E)$ where $L \cap R = \emptyset$ and every edge connects a vertex in $L$ to a vertex in $R$. You are also given a set of non-negative integer limits $\ell : (L \cup R) \to \mathbb{Z}_{\geq 0}$. Describe and analyze an algorithm that returns a maximum size subset of edges $E' \subseteq E$ such that each vertex $v$ is incident to at most $\ell(v)$ edges in $E'$. As usual, you may express your running time in terms of $V$ and $E$, the number of vertices and edges in $G$.

**Solution:** As in bipartite matching, we’ll create a flow network $G_0 = (V_0, E_0)$ with capacities $c : E_0 \to \mathbb{R}_{\geq 0}$. We start with $V_0 := L \cup R$ and $E_0 = \emptyset$. We add two more vertices $s$ and $t$ to $V_0$. For each edge $uv \in E$ with $u \in L$ and $v \in R$ we add $u \rightarrow v$ to $E_0$ and set $c(u \rightarrow v) := 1$. For each vertex $u \in L$, we add an edge $s \rightarrow u$ to $E_0$ and set $c(s \rightarrow u) := \ell(u)$. For each vertex $v \in R$, we add an edge $v \rightarrow t$ to $E_0$ and set $c(v \rightarrow t) := \ell(v)$. We compute a maximum $(s, t)$-flow $f^*$ in $G_0$ using Orlin’s algorithm. Finally, we return each edge $uv$ with $u \in L$ and $v \in R$ such that $f^*(u \rightarrow v) = 1$.

We’ll now prove the subset of edges $E'$ returned by the algorithm is optimal. First, let $E^*$ be the largest subset of edges respecting the vertex limits. Let $f$ be the following flow: For each edge $uv \in E$ with $u \in L$ and $v \in R$, we let $f(u \rightarrow v) = 1$ if $uv \in E^*$ and $f(u \rightarrow v) = 0$ otherwise. For each $u \in L$, we let $f(s \rightarrow u)$ equal the number of edges in $E^*$ incident to $u$ and define $f(v \rightarrow t)$ similarly for each $v \in R$. This assignment satisfies conversation constraints. Also, each vertex $v \in L \cap R$ is incident to at most $\ell(v)$ edges in $E^*$ so the flow satisfies the capacity constraints. The total amount of the flow leaving $s$ is the number of edges incident to vertices in $L$, so $|f| = |E^*|$.

Observe $f^*$ must be integral, meaning for each edge $u \rightarrow v$ with $u \in L$ and $v \in R$ has exactly 0 or 1 units of flow. Each vertex $u \in L$ has at most $\ell(u)$ units of flow on its incoming edges, so at most $\ell(u)$ of its outgoing edges contain flow. A similar statement holds for any $v \in R$, so $E'$ is a feasible solution to our problem. Finally, the $|E'|$ is equal to the number of edges containing flow that leave members of $L$, meaning it equals the total flow leaving $s$. Therefore, $|E'| = |f^*| \geq |f| = |E^*|$. We have that $|E'|$ is optimal.

Building the flow network, running Orlin’s algorithm, and returning the subset $E'$ takes $O(VE)$ time total.

**Rubric:** 10 points total. 6 points for the algorithm. 2 points for justification. 2 points for running time analysis.
Suppose you are given a flow network $G$ with integer edge capacities and an integer maximum flow $f^*$ in $G$. Describe algorithms for the following operations:

Both algorithms should modify $f^*$ so that it is still a maximum flow under the new capacities more quickly than recomputing a maximum flow from scratch.

(a) **Increment**: Increase the capacity of edge $e$ by 1 and update the maximum flow.

**Solution**: Let $G = (V, E)$ and $c : E \to \mathbb{R}_{\geq 0}$ be the directed graph and capacity function of our flow network. To perform $\text{INCREMENT}(e)$, we set $c(e) \leftarrow c(e) + 1$. Flow $f^*$ is still feasible after this change. We build the residual graph $G_{f^*}$ and find an augmenting path $P$ from $s$ to $t$ if one exists. If no path exists, then $f^*$ is still maximum and we are done. Otherwise, we update $f^*$ by pushing one unit of flow along $P$ and we are done.

Observe $e$ must be part of a minimum cut in order for the minimum cut capacity and maximum flow value to increase. The capacity of $e$’s cut increases by exactly one in that case, so pushing along a single augmenting path suffices to update $f^*$. **Building the residual graph and pushing along an arbitrary augmenting path takes $O(E)$ time total.**

**Rubric**: 1.5 extra credit points total. 1 point for the algorithm and 0.5 points for running time. No credit for an $\Omega(VE)$ time algorithm.

(b) **Decrement**: Decrease the capacity of edge $e$ by 1 and update the maximum flow.

**Solution**: Let $G = (V, E)$ and $c : E \to \mathbb{R}_{\geq 0}$ be the directed graph and capacity function of our flow network. If $e$ is not saturated by $f^*$ before the decrement, then $f^*$ is still feasible after the decrement. Decreasing capacities can only decrease maximum flow values, so $f^*$ must still be maximum and we are done.

Suppose instead that $f^*(e) = c(e)$. Let $u \to w = e$. We’ll assume $f^*(u \to w) = c(u \to w) \geq 1$ so we still have a flow network after decrementing. We build the residual graph $G_{f^*}$ (note we have not yet decremented $c(u \to w)$, so the residual capacities are defined in terms of the original given capacities). Note edge $w \to u$ is in the residual graph, because $f^*(w \to u) \geq 1$. We search for a path $P$ from $u$ to $w$ in the residual graph. If the path $P$ exists, we push one unit of flow along the cycle $P \circ w \to u$. The same arguments as in the text imply that the new flow is feasible and has the same value as $f^*$. We return the new flow.

Now suppose instead that no $u \to v$ path $P$ exists. We find a residual graph paths $P_1$ from $t$ to $v$ and $P_2$ from $u$ to $s$. Then we modify $f^*$ by pushing one unit of flow along $t$ to $s$ path $P_1 \circ w \to v \circ P_2$ and return the resulting flow.

We still need to prove the proposed path exists in the second case. Let $S$ be all the vertices reachable from $u$ in $G_{f^*}$, and let $T = V \setminus S$. Let $f'$ be the flow $f^*$ restricted to the edges with both endpoints in $S$. For any vertices $v_i \in S$ and $v_t \in T$, we have $f'(v_i \to v_t) = 0$. Therefore, only the flow on outgoing edges of any such $v_s$ has been reduced in $f'$. For any $v_s \in (S \setminus \{s\})$, we have $\partial f'(v) \leq \partial f^*(v) = 0$. In particular, $\partial f'(u) < 0$, because $f'$ no longer
sends one or more units of flow along $u \rightarrow w$. Finally, $\sum_{v \in S} \partial f'(v) = 0$, so we need $s \in S$ as it is the only possible vertex with positive outgoing flow. We conclude path $P_1$ exists. A similar argument implies $P_2$ exists. Walk $P_1 \circ w \rightarrow v \circ P_2$ is a path, because if $P_2$ contained a vertex $x$ of $P_1$, then there would have been a walk from $u$ to $x$ to $w$, contradicting there being no $u$ to $w$ path. Therefore, the algorithm is fine to push a unit of flow from $t$ to $s$ along the path, reducing the flow value by one.

Finally, $t \in T$, because otherwise there would be a residual graph path from $u$ to $t$ to $w$. Every edge leaving $S$ is saturated and every edge entering $S$ is avoided, so $(S, T)$ is a minimum $(s, t)$-cut. Reducing the capacity of $e$ by one reduces the minimum cut capacity by one, so the new flow is maximum for the new capacities.

The time to search for these paths and push flow is $O(E)$.

Rubric: 3.5 extra credit points total. 2 points for the algorithm, 1 point for justification, and 0.5 points for running time. No credit for an $\Omega(VE)$ time algorithm.