This homework contains three problems. Read the instructions for submitting homework on the course webpage.

Collaboration Policy: For this homework, each student can work in a group with up to three members. Only one solution for each group needs to be submitted. Follow the submission instructions carefully.

For problems that use maximum flows as a black box, a full-credit solution requires the following.

- A complete description of the relevant flow network, specifying the set of vertices, the set of edges (being careful about direction), the source and target vertices \( s \) and \( t \), and the capacity of every edge. (If the flow network is part of the original input, just say that.)

- A description of the algorithm to construct this flow network from the stated input. This could be as simple as “We can construct the flow network in \( O(n^3) \) time by brute force.”

- A description of the algorithm to extract the answer to the stated problem from the maximum flow. This could be as simple as “Return \( \text{True} \) if the maximum flow value is at least 42 and \( \text{False} \) otherwise.”

- A proof that your reduction is correct. This proof will almost always have two components. For example, if your algorithm returns a boolean, you should prove that its \( \text{True} \) answers are correct and that its \( \text{False} \) answers are correct. If your algorithm returns a number, you should prove that number is neither too large nor too small.

- The running time of the overall algorithm, expressed as a function of the original input parameters, not just the number of vertices and edges in your flow network.

- You may assume that maximum flows can be computed in \( O(VE) \) time. Do not regurgitate the maximum flow algorithm itself.

Reductions to other flow-based algorithms described in class or in the notes (for example: edge-disjoint paths, maximum bipartite matching, minimum-cost circulation) or to other standard graph problems (for example: reachability, minimum spanning tree, shortest paths) have similar requirements.
1. Finding median of a stream of numbers is not easy without using too much space. However, it is possible to find an approximate median of a stream. Consider a stream of elements $a_1, a_2, \ldots$ where element $a_i$ arrives at time $i$. Given a constant $\epsilon > 0$, an $\epsilon$-approximate median of the stream at time $t$ is an element whose rank is between $\lfloor(1/2 - \epsilon)t\rfloor$ and $\lceil(1/2 + \epsilon)t\rceil$ among elements of set $\{a_1, \ldots, a_t\}$. For simplicity let us assume that all elements in the stream are distinct, and are at most $n$ (note that this also implies $t \leq n$).

(a) Let $S$ be a set of $c$ elements sampled uniformly at random from set $\{a_1, \ldots, a_t\}$, and $0 \leq \delta \leq 1$. Let $X$ be a random variable that captures the number of elements in $S$ with rank strictly less than $\lfloor \delta t \rfloor$. Show that $\delta c - \frac{2c}{t} \leq E[X] \leq \delta c$ in case of both sampling with replacement and sampling without replacement.

(b) Given a constant $k > 0$ we want to find an $\epsilon$-approximate median of the stream with probability at least $(1 - \frac{1}{k})$ at any time $t$. Design a randomized algorithm to do the same. At any time $t$, your algorithm should be able to return an $\epsilon$-approximate median of set $\{a_1, \ldots, a_t\}$ with probability $(1 - \frac{1}{k})$. The goal is to do this using as less space as possible.

[Hint: Use part (a)]

2. Recall the CountMin sketch algorithm to estimate the frequencies of the items in a stream. Suppose the algorithm uses exactly one hash function that maps elements of the stream to $\{0, \ldots, (m-1)\}$ where $m = 10$. Give an example of an input stream $\sigma$, say of length $t$, such that the probability is very high that for at least one of the items $j \in \sigma$, the estimate of its frequency is much larger than its actual frequency. More precisely, give an example such that (for $t$ large enough) the probability that there is an item $j$ with $f'_j - f_j \geq t/2$ is at least 0.99, where $t$ is the stream length. Here $f'_j$ is the estimated frequency of $j$ from the sketch and $f_j$ is the true frequency. Note that element $j$ has to be part of the stream, and therefore has to appear at least once.

Recall that, assuming elements of the stream are from set $[1..n]$ the hash function $h : [1..n] \to [0..(m-1)]$ is chosen uniformly at random from a 2-universal hash family $\mathcal{H}$. That is for any $x, y \in [1..n]$, if $x \neq y$ then $Pr_{h \in \mathcal{H}}[h(x) = h(y)] = \frac{1}{m}$. Additionally, (if needed) assume that $\mathcal{H}$ is 3-uniform as well.

3. Suppose $f$ is an $s$-$t$ flow in a network $G = (V, E)$ and let $f'$ be another flow with value larger than that of $f$. Prove that there is an $s$-$t$ flow of value $\text{val}(f') - \text{val}(f)$ in the network $G_f$. 

2