For the following exercise, we need the following notions from complexity theory.

- We say problem $A$ is **deterministic poly-time reducible** to problem $B$ if there exists a deterministic algorithm $R$ for solving problem $A$ that makes calls to a subroutine for problem $B$, where the running time of $R$ (not including the running time for the subroutine for $B$) is polynomial in the input length.

- We say that $A$ and $B$ are **deterministic poly-time equivalent** if $A$ is deterministic poly-time reducible to $B$ and $B$ is deterministic poly-time reducible to $A$.

1. Consider the following problems.

   (a) Given a prime $p$, a prime $q$ that divides $p - 1$, an element $\gamma \in \mathbb{Z}_p^*$ generating a subgroup $G$ of $\mathbb{Z}_p^*$ of order $q$, and two elements $\alpha, \beta \in G$, compute $\gamma^{xy}$, where $x := \log_\gamma \alpha$ and $y := \log_\gamma \beta$. (This is just the Diffie–Hellman problem.)

   (b) Given a prime $p$, a prime $q$ that divides $p - 1$, an element $\gamma \in \mathbb{Z}_p^*$ generating a subgroup $G$ of $\mathbb{Z}_p^*$ of order $q$, and an element $\alpha \in G$, compute $\gamma^{x^2}$, where $x := \log_\gamma \alpha$.

   (c) Given a prime $p$, a prime $q$ that divides $p - 1$, an element $\gamma \in \mathbb{Z}_p^*$ generating a subgroup $G$ of $\mathbb{Z}_p^*$ of order $q$, and two elements $\alpha, \beta \in G$, with $\beta \neq 1$, compute $\gamma^{xy'}$, where $x := \log_\gamma \alpha$, $y' := y^{-1} \mod q$, and $y := \log_\gamma \beta$.

   (d) Given a prime $p$, a prime $q$ that divides $p - 1$, an element $\gamma \in \mathbb{Z}_p^*$ generating a subgroup $G$ of $\mathbb{Z}_p^*$ of order $q$, and an element $\alpha \in G$, with $\alpha \neq 1$, compute $\gamma^{x'}$, where $x' := x^{-1} \mod q$ and $x := \log_\gamma \alpha$.

Show that these problems are deterministic poly-time equivalent. Moreover, your reductions should preserve the values of $p$, $q$, and $\gamma$; that is, if the algorithm that reduces one problem to another takes as input an instance of the former problem of the form $(p, q, \gamma, \ldots)$, it should invoke the subroutine for the latter problem with inputs of the form $(p, q, \gamma, \ldots)$.

2. Suppose there is a probabilistic algorithm $A$ that takes as input a prime $p$, a prime $q$ that divides $p - 1$, and an element $\gamma \in \mathbb{Z}_p^*$ generating a subgroup $G$ of $\mathbb{Z}_p^*$ of order $q$. The algorithm also takes as input $\alpha \in G$. It outputs either “failure,” or $\log_\gamma \alpha$. Furthermore, assume that $A$ runs in expected polynomial time, and that for all $p$, $q$, and $\gamma$ of the above form, and for randomly chosen $\alpha \in G$, $A$ succeeds in computing $\log_\gamma \alpha$ with probability $\epsilon(p, q, \gamma)$. Here, the probability is taken over the random choice of $\alpha$, as well as the random choices made during the execution of $A$. Show how to use $A$ to construct another probabilistic algorithm $A'$ that takes as input $p$, $q$, and $\gamma$ as above, as well as $\alpha \in G$, runs in expected polynomial time, and that satisfies the following property:

   if $\epsilon(p, q, \gamma) \geq 0.001$, then for all $\alpha \in G$, $A'$ computes $\log_\gamma \alpha$ with probability at least 0.999.
The algorithm $A'$ in the previous exercise is an example of a **random self-reduction**, that is, an algorithm that reduces the task of solving an arbitrary instance of a given problem to that of solving a random instance of the problem. Intuitively, the existence of such a reduction means that the problem is no harder in the worst case than on average.

3. Let $p$ be a prime, $q$ be a prime dividing $p - 1$, and let $G$ be the subgroup of $\mathbb{Z}_p^*$ of order $q$.

   (a) We may define a hash function $H_1 : \mathbb{Z}_q \times \mathbb{Z}_q \rightarrow G$ as follows. Let $\gamma$ be a randomly chosen generator for $G$, and let $\alpha$ be a randomly chosen element of $G$. For $(x, y) \in \mathbb{Z}_q \times \mathbb{Z}_q$, define $H_1(x, y) := \gamma^x \alpha^y$. Here, $\gamma$ and $\alpha$ are public parameters that define the function $H_1$. Show that under the DL assumption for $G$, $H_1$ is collision resistant.

   (b) We may extend the input domain of $H_1$ as follows. Again, let $\gamma$ be a randomly chosen generator for $G$, and let $\alpha_1, \ldots, \alpha_k$ be randomly chosen elements of $G$. The group elements $\gamma, \alpha_1, \ldots, \alpha_k$ define the hash function $H_2(x, y_1, \ldots, y_k) := \gamma^x \alpha_1^{y_1} \cdots \alpha_k^{y_k}$. Show that under the DL assumption for $G$, $H_2$ is collision resistant.

4. You are given a secure signature scheme whose signing algorithm is probabilistic. Show how to use a PRF to convert the scheme into a secure signature scheme whose signing algorithm is deterministic (and the verification algorithm remains the same).