Problem 1

Assume that $f$ is a length preserving one-way function, i.e., for every $x \in \{0,1\}^*$ it holds that $|f(x)| = |x|$. For each of the following functions $g$, prove that $g$ is a one-way function, or provide a counterexample to demonstrate that it is not.

A: $g(x) = f(f(x))$
B: $g(x) = f(\bar{x})$
C: $g(x) = f(x) \oplus x$
D: $g(x, y) = f(x \oplus y)$
E: $g(x) = f(x) \parallel f(\bar{x})$

(Above $\bar{x}$ denotes the bitwise complement of $x$ and $\parallel$ denotes concatenation, e.g., $1011 \parallel 1011 = 10110100$.)

Problem 2

Prove that if one-way functions exist then $P \neq NP$.

Problem 3

Let $p$ be a prime and let $g$ and $h$ be (not necessarily distinct) generators of $\mathbb{Z}_p^*$. Prove or disprove the following statements:

A: $\{x \leftarrow \mathbb{Z}_p^* : g^x \mod p\} = \{x \leftarrow \mathbb{Z}_p^* : y \leftarrow \mathbb{Z}_p^* : g^{xy} \mod p\}$
B: $\{x \leftarrow \mathbb{Z}_p^* : g^x \mod p\} = \{x \leftarrow \mathbb{Z}_p^* : h^x \mod p\}$
C: $\{x \leftarrow \mathbb{Z}_p^* : g^x \mod p\} = \{x \leftarrow \mathbb{Z}_p^* : x^g \mod p\}$
D: $\{x \leftarrow \mathbb{Z}_p^* : x^g \mod p\} = \{x \leftarrow \mathbb{Z}_p^* : x^{gh} \mod p\}$

(Recall that $\{x \leftarrow \mathbb{Z}_p^* : g^x \mod p\}$ is a probability distribution. You are being asked to prove or disprove the statement that two probability distributions are identical.)

Problem 4

Suppose that you have a polynomial-time algorithm $A$ that solves the Discrete Logarithm Problem in a special case. Namely on inputs $p$, $g$, and $g^x \mod p$, the algorithm $A$ outputs $x$ if $p$ is a prime, $g$ is a generator of $\mathbb{Z}_p^*$ and $g^x \mod p$ is prime.

Show that there exists a probabilistic polynomial-time algorithm $B$ that solves any instance of the Discrete Logarithm Problem.
Problem 5

In this problem, we study how to efficiently sample generators modulo a prime.

Let \( p \) be a prime. The group \( \mathbb{Z}_p^* \) can be shown to be cyclic of order \( p - 1 \); in fact, while proving this, one also obtains the fact that the number of elements of order \( p - 1 \) in \( \mathbb{Z}_p^* \) (i.e., the number of generators in \( \mathbb{Z}_p^* \)) is equal to \( \phi(p - 1) \). Since \( \phi(n) = \Theta(n/\log \log n) \), the quantity \( \phi(p - 1)/p - 1 \) is non-negligible. In particular, by choosing an element \( g \) of \( \mathbb{Z}_p^* \) at random, the probability that \( g \) is a generator of \( \mathbb{Z}_p^* \) is non-negligible. However, given an element \( g \) in \( \mathbb{Z}_p^* \), how can we decide if it is a generator or not?

Describe a polynomial-time algorithm that, on input an element \( g \in \mathbb{Z}_p^* \), an odd prime \( p \), and the factorization of \( p - 1 \), decides whether \( g \) is a generator of \( \mathbb{Z}_p^* \).

(Note: Efficiently sampling generators modulo a prime is sometimes needed in practice, such as in Elgamal’s public-key cryptosystem. But, how does one obtain the factorization of \( p - 1 \)? Usually, one generates the prime \( p \) along with the factorization of \( p - 1 \). For example, in Elgamal’s public-key cryptosystem a prime \( p \) is chosen to have the form \( p = 2q + 1 \) for some prime \( q \), so that \( p - 1 = 2q \); a prime of this form is called a safe prime.)

Problem 6

Give a strategy to distinguish between \( (g^x, g^y, g^{xy}) \mod p \) and \( (g^x, g^y, g^r) \mod p \) with non-negligible advantage, where \( x, y, r \) are chosen at random such that \( 1 \leq x, y, r \leq p - 1 \), and \( g \) is a generator of \( \mathbb{Z}_p^* \).