Exercise 1: Sieve of Eratosthenes (4 points)

The Sieve of Eratosthenes is a simple algorithm to iteratively generate all primes up to some number $M$. It works as follows:

1. Create a list of the integers from 2 to $M$.
2. Initially, set $p$ to 2 (a prime).
3. Mark all integer multiples of $p$ except $p$ itself in the list; i.e., mark $2p, 3p, \ldots, \lfloor M/p \rfloor \cdot p$.
4. Set $p$ to the next higher unmarked number. If $p \leq \sqrt{M}$, repeat; if $p > \sqrt{M}$ or if there is no unmarked number left, stop.
5. Return the set of all unmarked numbers.

Prove that this algorithm returns exactly the subset of primes up to $M$. Also show how to use the Sieve of Eratosthenes to determine the first $n$ primes, and determine the complexity of this method in terms of $n$.

Exercise 2: Chinese remaindering for integers (4 points)

The Chinese remainder algorithm allows us to recover a non-negative integer $m$, with $0 \leq m < \prod_{i=1}^{k} p_i$, from the modular images $m \bmod p_i$. Describe a method to recover an integer $m \in \mathbb{Z}$ with $-\frac{1}{2} \prod_{i=1}^{k} p_i < m < \frac{1}{2} \prod_{i=1}^{k} p_i$ from the modular images $m \bmod p_i$ and give a proof.

Exercise 3: Modular determinant computation (4 points)

Develop a small primes modular algorithm to compute the determinant of square integer matrices. Analyze its running time in terms of $n$ and $\tau$ for input matrices of size $n \times n$ and with integer entries of bitsize bounded by $\tau$.

Exercise 4: Small primes polynomial GCD (4 point + 4 bonus points)

1. Give explicit bounds on the bitsize of the coefficients of $\gcd(f, g)$, where $f$ and $g$ are integer polynomials of degree bounded by $n$ with coefficients of bitsize bounded by $\tau$. Further, determine an explicit bound in $n$ and $\tau$ on the number of unlucky primes for such a pair $(f, g)$. (Recall that a prime $p$ is unlucky for the small primes GCD algorithm if $p \mid \text{LC}(f)$ or $p \mid \text{LC}(g)$ or $\deg \gcd(f, g) \neq \deg \gcd(f \bmod p, g \bmod p)$.)

2. (Bonus) Give a randomized Las Vegas-method with expected runtime $\tilde{O}(n \tau)$ for the computation of $\gcd(f, g) \in \mathbb{Z}[x]$ for $f$ and $g$ as above. That is, your algorithm must always give correct results, but it is allowed to take longer than expected (such as, e.g., quicksort).