You do not have to hand in these exercises, they are for your practice only.

1. **Finite fields** $\mathbb{F}_q$: In class, we discussed $\mathbb{F}_q = \{0, 1, \ldots, q - 1\}$, where $q$ is a prime and addition and multiplication is done modulo $q$.

   $\mathbb{F}_2$ is just a bit with addition modulo 2 (XOR) and the usual multiplication: $1 \oplus 1 = 0$, $1 \times 1 = 1$ etc. In mathematics, $\mathbb{F}_q$ is called a finite ‘field’ with $q$ elements.

   In $\mathbb{F}_q$, any nonzero number has a multiplicative inverse, i.e., if $x \neq 0$ is in $\mathbb{F}_q$ then there exists a unique element $y$ in $\mathbb{F}_q$ such that $xy = yx = 1$ (all arithmetic is done modulo $q$). We usually write $x^{-1}$ for this element $y$ and call it the inverse of $x$. For example, $2^{-1} = 2$ in $\mathbb{F}_3$, since $2 \times 2 = 4 \pmod{3} = 1$.

   (a) Write down all nonzero elements of $\mathbb{F}_7$ and find their inverses.

   In class, we said that an element $\alpha \in \mathbb{F}_q$ is called a generator (or ‘primitive element’) if $\{\alpha, \alpha^2, \ldots, \alpha^{q-1}\}$ runs over all nonzero numbers in $\mathbb{F}_q$. Generators exist for any prime $q$.

   (b) Find all generators of $\mathbb{F}_7$.

   **Remark:** The restriction to prime numbers is important. Otherwise, inverses and generators do not necessarily exist.

2. **Dividing polynomials**: Just like we can divide integers by each other when we are happy with leaving a remainder, we can divide any two polynomials with remainder. That is, given two polynomials $A$ and $B$, where $B \neq 0$, there are unique polynomials $Q$ and $R$ such that

   $$A = QB + R,$$

   and the degree of $R$ is less than the degree of $B$. We will call $Q$ the quotient and $R$ the remainder, and write $R = A \mod B$. You can compute $Q$ and $R$ in completely the same way how you do ‘long division’ between integers to figure out their quotient and remainder:

   ```
   Q <- 0
   R <- A
   while R and degree(R) >= degree(B):
       d <- degree(R) - degree(B)
       L <- leading_coeff(R) * leading_coeff(B)^{-1} * X^d
       Q <- Q + L
       R <- R - L * B
   ```

   Here, the leading coefficient of a polynomial $P = p_0 + p_1 X + \cdots + p_d X^d$ of degree $d$ is $p_d$. That is, we start with $A$ and repeatedly subtract a suitable multiple of $B$ such that the degree decreases. This algorithm works not only for polynomials whose coefficients are real numbers, but also when the coefficients are in $\mathbb{F}_q$.

   (a) Compute the quotient and remainder for the following polynomials with coefficients in $\mathbb{F}_3$: $A = X^3 + 1$ and $B = 2X$.  

   1
(b) Compute the quotient and remainder for the following polynomials with coefficients in \( \mathbb{F}_5 \): \( A = X^3 + 2X \) and \( B = X + 4 \).

3. **Reed-Solomon encoding:** Consider the Reed-Solomon code with parameters \( q = 7, N = 4, K = 2, \) and \( \alpha = 3 \).
   
   (a) Compute the generator polynomial \( G \).
   
   (b) Write down the codeword \([x_1, x_2, x_3, x_4]\) for a general message \([s_1, s_2]\) ∈ \( \mathbb{F}_7^2 \).

4. **Decoding erasure errors:** Imagine that a codeword \( x^N \) for a Reed-Solomon code is corrupted by \( C \) many erasure errors. That is, \( y^N \) differs from \( x^N \) at \( C \) locations and you know what these locations are. If \( C \leq T = N - K \), how can you decode the codeword? If this seems hard do not despair – we will discuss this on Thursday in class!

   *Hint:* Think of \( x^N \) and \( y^N \) as coefficients of polynomials \( M \) and \( R \). Then decoding is equivalent to figuring out the error polynomial \( E = R - M \), which has \( C \) unknown coefficients. Observe that \( E(\alpha) = R(\alpha), \ldots, E(\alpha^T) = R(\alpha^T) \). *Why does this help?*