§5.4 and “Notes on binary codes.” Understand that a binary code of length \(n\) is a subset \(C\) of the abelian group \(\mathbb{Z}_2 \times \ldots \times \mathbb{Z}_2\) (\(n\) times), also written as \(B^n\) (sequences of length \(n\) of 0s and 1s; addition is vector addition: componentwise, mod 2). The code \(C\) is **linear** if \(C\) is a subgroup of \(B^n\). (Since mod 2 every word is its own inverse, the condition amounts to \(C\) being closed under addition). Understand the (Hamming) distance p.234 between codewords and for a linear code know how to reckon the usefulness of a code (for error detection &/or correction), i.e. the minimum distance between codewords, by inspection of the set of code-words p.237. [After p.237 switch to material in the “Notes.”]

Interpreting codewords as vectors, understand how a homomorphism (a linear transformation) \(h : B^n \to B^m\) defines a linear code \(C_h\) by \(C_h = \{ x \in B^n | h(x) = 0 \}\), the set of all \(n\)-tuples which \(h\) sends to the zero \(m\)-tuple in \(B^m\). (This is called the **kernel** of \(h\)). Understand how \(h\) can be represented by right-multiplication by a matrix \(H\): the \(m\)-vector \(h(x)\) is the matrix product \(xH\); the first row of \(H\) is the \(m\)-vector \(h(1,0,0,\ldots,0)\), the second row is \(h(0,1,0,\ldots,0)\), etc.; we can call the code \(C_H\). Know how to deduce the error-detecting or error-correcting properties of \(C_H\) by inspection of the matrix \(H\).

§6.1, 6.2 Understand the similarity between the divisibility of polynomials \(s(x), t(x), \ldots\) with coefficients in a field (the real numbers, \(\mathbb{Z}_2\), etc.) and the divisibility of integers. Be able to carry out the division algorithm (“long division”) for polynomials, giving a quotient and a remainder. Be able to carry out the Euclidean Algorithm to calculate a greatest common divisor \(d(x)\) of \(s(x), t(x)\) and to write \(d(x)\) as a polynomial linear combination of \(s(x)\) and \(t(x)\). Note one difference: a polynomial \(p(x)\) has a linear factor \((x - \alpha)\) if and only if \(p(\alpha) = 0\). This is very useful in finite fields, since there are only finitely many possible \(\alpha\).

§6.3 Understand the definition of **irreducible** p.273; the distinction between irreducible and **prime** is not important in this context. Understand the proof of Theorem 6.3.4 (every polynomial can be written as a product of irreducibles) and the difference from Theorem 1.3.3 (unique factorization for integers): an irreducible factor is only determined up to a nonzero multiplicative constant. When the coefficient field is \(\mathbb{Z}_2\) this difference does not manifest itself since the only nonzero constant is 1. Understand Examples 1 and 2 on p.277 completely.

§6.4 Understand that polynomial congruence classes are defined, and have many properties like, congruence classes of integers **mod** \(m\). In particular, understand that when a polynomial \(p(x)\) is irreducible, every nonzero congruence class **mod** \(p(x)\) has a multiplicative inverse: Proposition 6.4.3, Example 2 p.281 and continuing in the Example on p.282. Be familiar with the examples worked out in class:

- in \(\mathbb{Z}_2[X]\) using \(p(x) = x^2 + x + 1\), \(p(x) = x^3 + x + 1\)
- in \(\mathbb{Z}_3[X]\) using \(p(x) = x^2 + 1\)
- in \(\mathbb{R}[X]\) using \(p(x) = x^2 + 1\)
- in \(\mathbb{Q}[X]\) using \(p(x) = x^2 - 2\)

Be able to calculate products and inverses of equivalence classes in these and similar cases.