MAT 312/AMS 351 Fall 2010 Review for Final *Revised 12/9/10.*

NOTE: Final is cumulative. Use review sheets 1 and 2 as well as this one. Also review all homework, as well as midterms 1 and 2.

§3.2 Given a permutation $\pi \in S(n)$, know how to compute its order (Definition 3.16). Understand how to apply Proposiiton 3.21 to calculate the order of a permutation which has been written as a product of *disjoint* cycles. *Exercise 2 p. 83.* Understand what it means for two permutations $\pi_1, \pi_2 \in S(n)$ to be conjugate (bottom of p.79); that congugacy is an equivalence relation on $S(n)$; understand what the *shape* of a permutation is (Definition 3.26) and that a *conjugacy class* (defined at top of p. 80) is made up of all the permutations with the same shape (Theorem 3.27). *Homework 9, Exercise 5.* Understand the *sign* of a permutation (Definition 3.31; alternate definition in "Notes and Exercises on Permutations and Matrices"). Know that a permutation with sign -1 is *odd*, and one with sign 1 is *even*. Understand that every *transposition* (Definition 3.4) has sign -1. Know how to prove Lemma 3.35 (every cycle is a product of transpositions) and understand why an odd-length cycle is an even permutation, and vice-versa. Be able to apply Theorem 3.36 (if a permutation is written as a product of transpositions, the permutation is odd if the number of those transpositions is odd, and even if that number is even). Understand that the order, the sign and the shape are the same for conjugate permutations.

§3.3 Understand the definition of a *group*. If you need examples, use the set $S(n)$ of permutations of n elements, with composition law the product of permutations. Examples derived from the integers, e.g. $\mathbf{Z}_n$ (with $+$) and $\mathbf{Z}_n^*$ (with multiplication) have a rich structure but are all *commutative*, so not typical. Make sure you are comfortable with the notation conventions (3.42). Be able to prove Theorem 3.43 (identity and inverses are unique).

§4.1 Understand the definition of the *order* of an element g of a group G. (Same definition as the order of a permutation). The concept of *subgroup* of G is elementary but important: it's a subset H of G which is closed under composition and inverses. In particular H, and the composition law of G applied to elements of H, constitute a group. Examples 4.11 (1) and (2). Special cases: for any element $g \in G$, the set of positive and negative powers of g, i.e. $\{g, g*g, g*g*g, \ldots, g^{-1}, g^{-1}*g^{-1}, \ldots, g*g^{-1} = e\}$ forms a subgroup, called the *cyclic subgroup* generated by g, and written $\langle g \rangle$ (Proposition 4.14). *Exercises (1), (2), (3) p. 98.*

§4.2 For a subgroup H of a group G, understand that the *left H-coset* corresponding to an element $a \in G$ is $aH = \{ah | h \in H\}$. Know that two left H-cosets either coincide or are disjoint, and that left-multiplication by a produces a 1-1 correspondence between H and aH. (Use Propositions 4.19 and 4.21, or follow the argument in "Notes and exercises on normal groups and quotient groups.") Understand that it follows, for a finite group G (i.e., a group with a finite number $o(G)$ of elements), that $o(G)$ must be an integer multiple of $o(H)$. This is Lagrange's Theorem (4.22).

§4.3 We did not cover homomorphisms in general, but only *isomorphisms*. These are contained in Definition 4.26: a function $\theta : G \to H$ from one group to another is an isomorphism if it is 1-1, onto and respects the group operations. So if $*$ is the operation in G, and @ is the operation in H, then $\theta(g_1 * g_2) = \theta(g_1)@\theta(g_2)$. Be able to show that $\theta^{-1} : H \to G$ is then also an isomorphism, and understand that isomorphic groups are "the

same group" with different labellings of the elements and the operation.

§4.5 (See also "Notes and exercises on normal groups and quotient groups"). Understand what *right cosets* are, and that in general, for a subgroup H of a non-commutative group G, the left coset aH and the right coset Ha are different subsets of G. (Example 2 of "Notes ..." also on p. 107, bottom). The subgroup H is called *normal* if $aH = Ha$ for every $a \in G$. (So in a commutative group, every subgroup is normal). Understand why if $o(H) = \frac{1}{2}o(G)$ then H must be normal. Understand why if H is normal, there is a well-defined multiplication between H-cosets, which makes the set of H-cosets into a group called the *quotient group* of G modulo H, and written G/H. This is Theorem 4.44, or Proposition 2 in the "Notes ...". Example, and Exercise 2, in the "Notes ... ".

§6.2 In this section, $\mathbf{B}$ stands for the group $\{0, 1\}$ with addition mod 2, and $\mathbf{B}^n$ stands for the group of n-tuples (n-component vectors) of elements of $\mathbf{B}$, with componentwise addition. A *code* is a function $f : \mathbf{B}^m \to \mathbf{B}^n$ which is 1-1 so it can be decoded. (This implies $m \leq n$). f is a *group code* if it preserves addition: $f(w_1 + w_2) = f(w_1) + f(w_2)$. Understand the *Hamming distance* d between vectors ("words") in $\mathbf{B}^n$: $d(w_1, w_2) =$ the number of places in which the two vectors have different components. Understand also the *weight* of a vector in $\mathbf{B}^n$: the number of "1"s among its components; and that the weight of w is equal to $d(w, (0, ..., 0))$, the distance between w and the zero-vector. Be able to prove that for a group code, the minimum distance between different words is equal to the minimum weight of a non-zero word. (Theorem 6.12).

A *code word* for the code $f : \mathbf{B}^m \to \mathbf{B}^n$ is $f(w)$ for some $w \in \mathbf{B}^m$. Be able to prove Theorems 6.7 and 6.8 about such a code f: it can detect k or fewer errors if and only if the minimum distance between different code words is $k + 1$; and it can correct k or fewer errors if and only if the minimum distance between different code words is $2k + 1$.

Understand how an $m \times n$ matrix G can be used to define a code f_G, by matrix multiplication: $f_G(w) = wG$, and that such a code is automatically a group code. Understand the concept of a *check bit*: In a word $c = (x_1, \ldots x_n)$, x_i is a (parity) check bit for locations $a_1, \ldots a_k$ if $x_i + x_{a_1} + \cdots + x_{a_k} = 0$. In other words, $x_i = 1$ if there is an odd number of "1"s in locations $a_1, \ldots a_k$, and $x_i = 0$ if that number is even. Understand also that position i in a code-word is a check bit for locations $a_1, \ldots a_k$ (in the code generated by matrix G) if the i-th column of G has "1"s in rows $a_1, \ldots a_k$, and "0" otherwise. Example at bottom of p. 137: here $m = 4$ and $n = 7$. Position 5 is a check bit on locations 2, 3, 4. Position 7 is a check bit on locations 1, 3, 4.

Hamming codes will *not* be covered on the final exam.