This is a review sheet/study guide for the final exam. There will be a review session Thursday at 1:00 in BE 115.

Before the midterm we covered chapters 9, 10, and 11 in the book. Any of the material in those chapters may be on the final. The topics include:

1. Turing Machines, transition tables, transition diagrams, configurations, accepting languages, Church-Turing thesis.

2. Computing functions with Turing Machines, how to handle multiple arguments.

3. Combining Turing Machines and standard subroutines (insert, delete, copy, concatenate, addition, etc.).

4. Variations on Turing Machines: Multiple tapes better? Two way tapes better? Need S (stay)? Need to write ∆? Is a finite tape OK?

5. Non-deterministic Turing Machines and how to “simulate” them with deterministic Turing Machines.

6. Universal Turing Machines, “standard” encoding of Turing Machines, letters, and strings.

7. The Recursive and Recursively Enumerable languages, and their basic definitions ($L$ accepted by TM, TM computes characteristic function for $L$).

8. Enumeration characterization of Recursive and Recursively Enumerable languages.


11. Countably infinite versus uncountable sets.

12. Not all languages Recursively Enumerable – counting argument and diagonalization.

13. Languages NSA and SA.

14. Unrestricted Grammars and how they can be “programmed”

15. Proofs that the languages generated by unrestricted grammars are exactly the Recursively Enumerable languages.

16. Regular grammars, regular languages, and finite automata.


18. Chomsky Hierarchy, and some recursive languages are not context sensitive.
After the midterm we covered chapters 12 through 15 in the book, although some topics (especially parts of chapter 13) were covered in less detail. The particular topics include:

1. Encodings and the relationships between decision problems and languages, decidable and undecidable problems/languages.
2. Reductions between problems: computable reduction functions, if-and-only-if property, \( L_1 \leq L_2 \) notation.
4. Many examples of reductions and undecidable problems.
5. Rice’s Theorem.
6. Post’s Correspondence Problem (PCP), Halting reduces to MPCP which reduces to PCP. This involves encoding halting computations of a TM with correspondences of an MPCP instance.
7. ★ Undecidable grammar problems: some reductions from PCP, halting computations as intersections of 2 CF languages, etc.
8. Computable and Uncomputable functions, Busy beaver function.
9. ★ Initial functions, primitive recursive functions, bounded quantification is primitive recursive.
10. ★ Bounded minimalization, unbounded minimalization and the \( \mu \)-recursive functions.
11. ★ Gödel numbering and encoding of TM configurations as a number.
12. Computational complexity theory, time and space resources for TMs and non-deterministic TMs as a function of input size.
13. Complexity classes: \( \text{Time}(f(n)) \), \( \text{Space}(f(n)) \), \( \text{NTime}(f(n)) \), \( \text{NSpace}(f(n)) \).
14. Why \( \text{Time}(f(n)) \) where \( f(n) < 2n + 2 \) doesn’t make sense, and using a read-only input tape for \( \text{Space}(f(n)) \) where \( f(n) < n \).
15. “Normal” or “nice” \( f(n) \): the step counting functions.
16. Speedup theorems (we emphasized time, but Space, NTime, NSpace, also have speedup theorems).
17. Diagonalization and hierarchy Theorems: \( \text{Time}(f(n)) \) is smaller than \( \text{Time}(Cn^2 f(n)^2) \).
18. Containments between complexity classes, e.g. \( \text{Time}(f(n)) \) is contained in \( \text{Space}(f(n)) \), \( \text{Space}(f(n)) \) is contained in \( \text{Time}(Cf(n)) \), etc.
19. Savitch’s theorem: simulating Non-deterministic space with deterministic space.
20. Optimization problems, related decision problems, “reasonable” encodings, and languages. Using a subroutine for the decision problem to solve the related Optimization problem.
21. “Tractable” problems, definitions of $\mathcal{P}$ and $\mathcal{NP}$, closure and robustness properties of $\mathcal{P}$, showing that a problem is in $\mathcal{NP}$.

22. Polynomial time reductions, and their properties and consequences (for example, $L_1 \leq_P L_2$ and $L_2 \in \mathcal{P}$ implies $L_1 \in \mathcal{P}$).

23. Definition of $\mathcal{NP}$-completeness. Why showing that a problem is $\mathcal{NP}$-complete is “evidence” (but not a proof) that the problem is not in $\mathcal{P}$.

24. CNF-satisfiability is $\mathcal{NP}$-complete (Cook’s Theorem).

25. Basic $\mathcal{NP}$-complete problems and the reductions showing that they are $\mathcal{NP}$-complete: 3-Sat, Clique, Vertex Cover, 3-colorability (or $k$-colorability), subset sum.

26. $\star$ Co-$\mathcal{NP}$ and the possible “states of the world.” Borderline problems: Primality and Graph Isomorphism.

27. $\star$ Dealing with $\mathcal{NP}$-completeness: Heuristics and approximation algorithms (if time permits).

Those sections marked with a $\star$ were covered quickly. Although they may appear on the final in short-answer or true-false questions, there will not be a long (i.e. proof) question on those topics. The exam will be comprehensive, and I expect that about 1/4 of the points will be on material covered before the midterm.