Topics for Exam 2

- Open books and notes but no electronic aids
- #s in “()” refer to homework problems; [] = Exercises

- Chap. 2.1 Context Free Grammars (p.102)
  - [2.4, 2.6, 2.9, 2.28] \(4.1,4.3b\) Create formal description of a CFG from language description (p.104)
  - Describe a language given a CFG
  - [2.1] Given a CFG and a string: show either/both a parse tree and/or a derivation
  - Know different kinds of derivations (esp. left-most)
  - (5.2b) Know/use/prove rules about combinations of CFLs
    - [2.16] Closed under U, Concat, *
    - [2.2] (5.3b,c) Not closed under ∩, complement
    - [2.18a] (5.3a) CFL \(∩\) RE = CFL
  - [2.23,24] Also (4.3.a) Show two set descriptions of a CFL are equal

- Chap. 2.2 Push Down Automata (p.111)
  - Understand formal definition of PDA (p. 111)
  - Understand role of \(\varepsilon\)s in transition rules (p. 114)
  - [2.5] (4.1,4.3b,5.2a) Create formal description of a PDA from a language description
  - [2.11] (4.2a) Create a formal description of a PDA from a CFG (pp. 119-120)
  - Given a PDA description and a string, show a derivation sequence
  - [2.12] (4.2b) Given a PDA, construct CFG (Lemma 2.27) (p. 122)

- Chap. 2.3 Non CFG Languages (p.125)
  - [2.34] Be able to estimate a pumping length from parameters of a CFG (p127)
  - [2.30-33] (5.1) Apply CFL pumping lemma to show a language is not CFL (p.126)

- Chap. 3.1. Turing Machines (p. 165)
  - Understand formal definition of TM (p. 168)
  - [3.1,2] Be able to specify configurations a TM goes thru during its computation, esp. accepting and rejecting (p. 1698)
  - Understand differences between formal, implementation, hi-level (p. 185)
  - (6.1a) Write formal description of TM from language description (p. 171-174)
  - [3.8] (6.1b) Write implementation description of TM from language description
  - Understand difference between a recognizer and a decider (p. 170)
  - Be able to define both an informal and a formal TM for either a decider (accept or reject) or a computation (e.g. add)
• Chap. 3.2. Variants of TMs (p. 176)
  • Understand variations of TMs and what transition rules for them look like
    • TM that can stay in place
    • Multiple tapes (p. 177)
    • [3.11] (6.2) Infinite in both directions
    • Nondeterministic (p. 178)
    • [3.10] Write-once TM
    • [3.11] Left reset TM
  • 3.6] Understand concept of a TM enumerator (p. 180)

• Chap. 3.3. Algorithms and TMs (p. 182)
  • Understand relationship between TMs and algorithms
  • Understand the Church-Turing thesis (p. 183)
  • Understand how simple graph problems might be computed by TMs (p. 185)

• Other
  • (6.3) Alternative computational models
  • [3.15] decidable languages closed under U, concat, *, ~, ∩
  • [3.16] Turing-recognizable languages closed under U, concat, *, ∩, homomorphism