CS311 Computational Structures
Spring 2020 Syllabus

Instructor:
Katie Casamento
cas28@pdx.edu
Office hours: Monday/Wednesday 3-4pm or by appointment (Zoom link on D2L)

Slack channel:
#cs311spring2020 (https://pdx-cs.slack.com/archives/C010HV51USJ)

Lecture:
Tuesdays/Thursdays 4:40-6:30pm (Zoom link on D2L)
Live ("synchronous") attendance optional
Lecture recordings available on D2L within 24 hours of live lectures

Course description:
Introduces the foundations of computing. Regular languages and finite automata. Context-free languages and pushdown automata. Turing machines and equivalent models of computation. Computability. Introduction to complexity. An appropriate programming language is used for programming experiments.

Course Goals:

Upon the successful completion of this course students will be able to:

1. Find regular grammars and context-free grammars for simple languages whose strings are described by given properties.
2. Apply algorithms to: transform regular expressions to NFAs, NFAs to DFAs, and DFAs to minimum-state DFAs; construct regular expressions from NFAs or DFAs; and transform between regular grammars and NFAs.
3. Apply algorithms to transform: between PDAs that accept by final state and those that accept by empty stack; and between context-free grammars and PDAs that accept by empty stack.
4. Describe LL(k) grammars; perform factorization if possible to reduce the size of k; and write recursive descent procedures and parse tables for simple LL(1) grammars.
5. Transform grammars by removing all left recursion and by removing all possible productions that have the empty string on the right side.

6. Apply pumping lemmas to prove that some simple languages are not regular or not context-free.
7. State the Church-Turing Thesis and solve simple problems with each of the following models of computation: Turing machines (single-tape and multi-tape); while-loop programs; partial recursive functions; Markov algorithms; Post algorithms; and Post systems.
8. Describe the concepts of unsolvable and partially solvable; state the halting problem and prove that it is unsolvable and partially solvable; and use diagonalization to prove that the set of total computable functions cannot be enumerated.
9. Describe the hierarchy of languages and give examples of languages at each level that do not belong in a lower level.
10. Describe the complexity classes P, NP, and PSPACE.

Required textbook:
Introduction to the Theory of Computation, Michael Sipser, 2012

Lecture schedule: (subject to change)

Week 1: Tuesday, March 31	Course overview Reading: Chapter 0
Week 1: Thursday, April 2	Languages and language construction Reading: Chapter 0
Week 2: Tuesday, April 7	Introduction to DFAs and regular languages Reading: Sections 1.1, 1.2
Week 2: Thursday, April 9	Closure properties of DFAs NFAs, DFA and NFA equivalence
Week 3: Tuesday, April 14	Regular expressions Limits of regular languages Reading: Sections 1.3, 1.4
Week 3: Thursday, April 16	Non-regular languages Pumping lemma
Week 4: Tuesday, April 21	Introduce context-free grammars Context-free grammars Reading: Sections 2.1, 2.2
Week 4: Thursday, April 23	Closure properties of context-free languages

	Ambiguity
Week 5: Tuesday, April 28	Pushdown automata Chomsky normal form Reading: Sections 2.3, 2.4
Week 5: Thursday, April 30	Non-context-free languages Context-free pumping lemma
Week 6: Tuesday, May 5	Finish context-free languages Introduce Turing machines
Week 6: Thursday, May 7	Turing machines Church-Turing thesis Reading: Section 3.1
Week 7: Tuesday, May 12	Turing machine variants Reading: Sections 3.2, 3.3
Week 7: Thursday, May 14	Non-deterministic Turing machines
Week 8: Tuesday, May 19	Limitations of Turing machines Reading: Chapter 4
Week 8: Thursday, May 21	Decidability Undecidability
Week 9: Tuesday, May 26	More decidability and undecidability Halting problem
Week 9: Thursday, May 28	Mapping reductions Rice's theorem Reading: Chapter 5
Week 10: Tuesday, June 2	Introduction to complexity P vs. NP
Week 10: Thursday, June 4	NP-complete problems Time complexity vs. space complexity