CIS 301 – Theory of Computation
Fall 2022

Textbooks*:** Required: "Introduction to the Theory of Computation 3rd Edition"***
by Michael Sipser ISBN-13: 978-1133187790 ISBN-10: 113318779X
*Recommended: "**Automata, Computability, and Complexity**" by Elaine Rich.*

Instructor: Dr. Supraja Gurajala
Phone: 315-267-2091
Email : gurajas@potsdam.edu

Office Hours: MWF: 10:00 am - 12:00 noon, Dunn Hall 307
Class Time/Place: MWF 9:00am - 9:50 am, Dunn Hall 202
Final Exam: Friday, December 16, 8:00 – 10:00 a.m.

Course Description:

Theory of Computation1, Regular and context-free languages, Turing machines, and the halting problem. Prerequisites: CIS 203, CIS 300.

Course Overview:

This course examines the theory underlying how computers work. A "cocktail-party" explanation of the content of this course is that we will attempt to describe the very simplest computer that does interesting computations and examine the range of computations it can do. This will require us to formally define what a computation is, what computations are "interesting", and then, finally, what a simple computer is.

We begin by examining the formal definition of a **language** and how computation can be modeled by deciding what strings are in (or out) of a given language. Consider, for example, addition: the string "1+1=2" is in the language of addition; "2 + 2 = 7" and "aardvark" are not. Computation is the execution of a decider; a binary, digital, general-purpose computer is an electronic decider (exact method of operation to be determined).

A hierarchy of different **classes** of languages have been identified. This course focuses on the regular, context-free, and decidable languages. These classes have wide applicability across computer science: regular languages in lexical analysis and text search; context-free languages in parsing programming languages; decidable languages in describing what is and **what is not** computable. Proving that a given language is (or is not) in a particular class is useful in distributed systems, parallel programming, programming language design, and compiler implementation.

This course will focus on grammars and automata and their use in classifying languages. There will also be an examination of reducibility and computability in terms of languages.

Learning Objectives:

Students who have taken this course should be able to gain:

- knowledge of discrete and continuous mathematics – including elementary probability and statistics – and the ability to apply logic and mathematical proof techniques to computing problems.

This course covers **languages**, sets of **strings** which are sequences of **characters** draw from some set of characters. These mathematical structures are the basis of the theory of computation. Students will be able to describe languages in multiple ways. Students will be able to write proofs using sets of strings and sequences of characters.

- knowledge of basic theory of computability and complexity of computation.

  Students will be able to construct language deciders from a description of a language, describe the language accepted by a given decider, and prove languages are or are not of a particular class of languages. Students will be able to define and apply computability, decidability, and acceptability in terms of automata and languages.

- knowledge of fundamental data structures and algorithms – including analysis of their correctness and complexity – related to various fields of computer science, and the ability to apply this knowledge to problems through the use of appropriate programming languages.

  Students will be able to define **algorithm** in terms of Turing machines. Students will be able to define what it means to be computable. Students will be familiar with proof by reduction.

Learning Outcomes:

Upon completing this course, students should be able to

1. Build a deterministic finite automaton (DFA), non-deterministic finite automaton (NFA), and/or regular expression (RE) for a given regular language.

2. Prove the equivalence of DFA, NFA, and RE in the set of languages they can accept; prove standard closure properties for regular languages.

3. Prove that a given language is/is not regular (apply regular pumping lemma).

4. Describe the language accepted by a given DFA, NFA, or RE.

5. Build a push-down automaton (PDA) and/or context-free grammar (CFG) for a given context-free language.

6. Prove the equivalence of PDA and CFG in the set of languages they can accept; prove standard closure properties for context-free languages.

7. Prove that a given language is/is not context-free (apply context-free pumping lemma).

8. Describe the language accepted by a given PDA or CFG.

9. Build a Turing machine (TM) for a given language.

10. Describe the language accepted by a given TM.

11. Describe the relationship between regular, context-free, and recursively-enumerable languages.

12. Describe the universal TM, its operation, and how it connects to physical computers.

13. Define and apply the term algorithm.

14. Connect and discuss the relationship between computable, decidable, and recognizable in terms of TM.

Grading for the Course:

1. ***Weekly Quizzes: 15%***

   A ten-minute weekly quiz will be given starting the second week of classes. It will be based on lectures and exercises (at the end of each chapter).

2. ***Homework / Programming Assignments: 25%***

   Several programming assignments and homeworks will be given based on the concepts discussed in lectures. These programming assignments and homeworks will be the essential part of the course. Programming assignments and homeworks will be posted on the moodle page along with the due date. Late assignments are penalized at 20% per calendar day that they are late. Your final submitted assignment should represent your individual work; it is, however, acceptable to discuss the solution approach with other students. You will be responsible for keeping track of due dates posted on moodle. Assignment submission policies are detailed in a separate section of this document.

3. ***Exams: 60%***

   a. Exam 1 – 14%
   b. Exam 2 – 14%
   c. Exam 3 – 14%
   d. Final Exam – 18%

   Exam will be closed book and closed notes. Any request for re-grading must be received in writing and within 3 days of receiving your graded exam back. Prior notice must be given to your instructor in case of emergency. No make-ups will be granted unless satisfactory documentation is produced to show an extenuating circumstance.
   Final exam is cumulative and is on Thursday, December 16, 12:30 – 2:30 pm

At the end of the semester I will calculate what fraction of the possible points you have earned, and your grade may be based on this distribution:

90% >=	A
80% - 90	B
70% - 80	C
60% - 70	D
< 60%	F

Note that final grades are determined using a class curve of the course-grade averages.

Course Policies

1. **Late work**

   All due dates for the course will be strictly enforced. Prior approval will be required from the instructor for any late submission, including making up missed exams.

2. **Attendance**

   Attending all lectures and labs and completing required work is crucial to your success in this course. While attendance is not graded *per se,* in-class graded work cannot be made up without prior arrangement with the instructor. In the event of absences from weekly labs, you are required to complete the missed lab work before the beginning of the next lab session. The instructor and CS tutors will be available to help you with completing labs during posted office and tutoring hours.

3. **Absences**

   As noted above, in-class graded work cannot be made up without prior arrangement with the instructor.

   Accommodation of Religious Observances: I will make reasonable accommodation for a student's

religious beliefs. Please notify me within the first week of classes about any scheduled class date that conflicts with a religious observance.

4. **Academic Integrity**

   You are expected follow the "SUNY Potsdam Academic Honor Code" (SUNY Potsdam Undergraduate Catalog, https://catalog.potsdam.edu/content.php?catoid=7&navoid=566) by doing your own work on all required work for the course unless specifically directed otherwise by the professor. **Copying is strictly forbidden, regardless of the source** (online, other students). Students caught cheating will receive a grade of 0 for that evaluation. More than one offense will result in dismissal from the course and possible disciplinary sanctions by the university. Academic Misconduct definitions, procedures, due process, and student rights are described on page in the SUNY Potsdam Undergraduate Catalog, as cited above.

5. Grade Distribution

   At the end of the semester, I will calculate what fraction of the possible points you have earned, and your grade will be based on this distribution:

   4.0: 95 – 100%
   3.7: 90 – 94%
   3.3: 85 – 89%
   3.0: 80 – 84%
   2.7: 77 – 79%
   2.3: 73 – 76%
   2.0: 70 – 72%
   1.7: 67 – 69%
   1.3: 63 – 66%
   1.0: 60 – 62%
   0.0: <60%

   Note that final grades may be determined using a class curve of the course-grade averages.

Tentative Schedule

Week 1	Introduction, Mathematical Notions and Terminology, Definitions,
Week 2	Theorems and Proofs , Regular Languages, Finite Automata
Week 3	Non deterministic Finite Automata
Week 4	Regular Expressions, Review & Midterm 1
Week 5	Non Regular Languages, Context Free Grammar
Week 6	Pushdown Automata, Non context Free Grammar
Week 7	Deterministeic Context Free Languages, Review & Midterm 2
Week 8	Turing Machines Variants of Turing Machines
Week 9	Definitions of Algorithms, Decidable Languages
Week 10	Undecidability, Review & Midterm 3
Week 11	Reducibility

Week 12	Undecidable problems
Week 13	Simple undecidable Problem
Week 14	Advance topics in Computability Theory
Week 15	Recursion & Review