CPS 440: Theoretical Foundations of Computer Science

Instructor	Ethan McGee
Office	AL 70 (3rd Floor Left Hall)
Email	emcgee@bju.edu
Phone	864.201.2635 (texting acceptable)
Office Hours	**MWThF:** 12:00 - 1:00 **T:** Electronic (Others by Appointment: calendly.com/ethantmcgee)

Text(s)	*Introducing the Theory of Computation 3rd ed.* (Sipser)
Meets	**TTh**: 1:30 - 2:45 (AL 314)
Credits	3
Prerequisites	CpS 210, Ma 150

Course Description

A study in finite state machines, Turing machines, computability and formal languages.

Course Context

This course serves as a major intersection between Computer Science and Mathematics, therefore, it fulfills the following objectives from each department.

Computer Science

- CS1. Design and implement efficient solutions to problems in various domains.
- CS2. Demonstrate understanding of fundamental concepts in computer science.
- CS3. Communicate technical information effectively, including software design and requirements documents.
- CS4. Evaluate and assess software technologies for use in solving specific problems.
- CS5. Apply Biblical principles of ethics to computing.

Math

- DM1. Mature the student in the theory and applications of mathematics.

- DM2. Provide the student the required mathematical background to function and contribute effectively in today's technological society
- DM3. Provide the student a platform for continued learning and development of his God-given abilities.
- DM4. Instill in the student a desire to use his abilities in service to Christ.
- DM5. Provide an appropriate liberal arts complement to a wide variety of majors

Course Goals

Specifically, the goals of this course are to:

- introduce the student to automata theory
  - study techniques and proofs with finite automata
  - study techniques and proofs with push-down automata
  - study techniques and proofs with turing machines
- use the concepts of automata theory to introduce complexity theory
  - determine a problem's complexity class
  - prove that certain complexity classes are subsets of larger complexity classes
  - prove that certain complexity classes are actually equal
- use the concepts of automata theory to introduce computability theory
  - determine a problem's computability
  - show fundamental limitations of computation

Assignments

Tests: taken in class; closed-book/notes unless otherwise announced.

Homeworks: almost weekly assignments given to re-inforce concepts covered in class.

Presentation: students will select and present a NP-Complete problem of their choice. The presentation will introduce the problem as well as detail the proof of the problem's NP-Completeness. Time limit for the presentation is 20 minutes.

Grading

Assignment Type	#	Points	Total Points
Tests	4	125	500
Homework	11*	25	250
Presentation	1	50	50
		Total	800

* Lowest Grade Dropped

Grade	Minimum	Maximum
A	89.5	100
B	79.5	89
C	69.5	79

Grade	Minimum	Maximum
D	59.5	69
F	0	59

Classroom Deportment

I realize this is college, and almost everyone is perpetually exhausted. If you find yourself falling asleep, feel free to stand up and walk to the back of the class to help yourself stay awake. If you need to go to the restroom, please exit to the rear of the class to avoid disturbing others. Please refrain from extended conversations in class so as not to disturb your fellow classmates. Asking your seatmate what is missing from your notes / example is fine; discussing lunch / non classroom topics is not.

Please be respectful of others around you while in class.

Deadlines / Late Work

Departmental Late Policy: https://cs.bju.edu/academics/policies/late-work-policy/

The instructor reserves the right to change assignment due dates as deemed necessary. Assignments are due, electronically, by 11:59 pm of the date posted in the course schedule unless otherwise noted.

Each student is given 1 free late waiver that allows them to turn in one assignment, at their discretion, up to 1 week after the deadline with no penalty. In order to use the free late, the student must 1) notify the professor before the deadline that they intend to use the free late, 2) describe the progress that they have made towards completion of the assignment and 3) provide an approximate date of delivery. Due to grading constraints during finals week, the professor reserves the right to shorten the late period for end of semester projects.

Getting Help

Students struggling with an assignment or concepts in the class are encouraged to ask the instructor for assistance either:

- in class
- before / after class
- during office hours
- via email
- via Teams (I typically make myself available via Teams to answer questions M-F from 9:00 pm to 10:30 pm)

In order to maximize your opportunity to receive help and receive the best possible grade on an assignment / in the course:

- Start assignments early. This will give you more opportunities to realize you don’t fully understand a concept and ask for assistance.
- Don’t wait until the night before an assignment is due to ask questions. The night an assignment is due typically sees a mad rush of questions, and I answer questions in the order that I receive them. There is no guarantee that I will be able to answer your question before the submission deadline.

- Request feedback. I cannot tell you what grade I would give to your particular solution for an assignment, but I can offer comments for how your solution can be improved.

Handbook Policies

Compliance with student handbook policies is expected during class.

Accommodations

Students needing accommodations due to a learning disability (visual, auditory, etc.) should provide an accommodation form obtained from the Academic Resource Center as soon as possible. Accommodations cannot be given without a form provided by the Academic Resource Center.

Academic Honesty and Integrity Policy

Cheating on assignments and tests is forbidden. All work is to be done individually unless group work is explicitly permitted. No collaboration is allowed on tests. For regular individual assignments, we expect that the submitted work represents the student's own intellectual effort, defined as follows:

1. The program was written primarily by the student. This means that most of the code (aside from starting code provided by the instructor) must have been crafted, not copied, by the student.
2. External resources used, whether electronic or from another human, must be documented as follows:
   - Code snippets copied from online resources must be documented by a comment just above the copied snippet giving the URL of the page containing the source.
   - Explanatory help or advice regarding the design or implementation of the solution received from people other than the instructor must be documented in a report accompanying the assignment submission. This report must detail:
     - Source of information (e.g., name/email of the person who helped)
     - Relevance (i.e., how this resource helped and/or what it provided)
   - Note that students must not consult a solution to the assignment as a resource in crafting their own solution, nor share their own solution with another student. Doing so constitutes cheating.
3. The student must be able to explain, on demand, the entirety of the program on both the syntactic and semantic level.

Not all kinds of programming assignments require the same demonstration of personal intellectual effort. In the absence of any specific instructions, students should assume that at a minimum:

- For individual lab assignments, requirements 1 and 3 apply.
- For individual programming assignments, all three requirements apply.
- For group programming projects, only requirement 3 applies.

Failure to comply with any relevant integrity requirement constitutes cheating. Such incidents will be reported to the academic integrity committee. To avoid trouble:

- Do not look at another student's program code when seeking assistance. On the other hand, if another student is seeking help from you, never use your own program code as an example. The only acceptable reason another student may look at your code is to help you find a problem in your program.

- Do not write program code while another student (or lab assistant) is sitting with you. You may work out designs in pseudocode on paper with another student, but you must write program code by yourself.
- When seeking assistance from another person on a program assignment, always get his/her name so you can fulfill the documentation requirements.

Copyright Policy

Copyright 2022⁄2023 Ethan McGee as to this syllabus and all lectures. Students are prohibited from selling (or being paid for taking) notes during the course to, or by any person, or commercial firm without the express written permission of the professor teaching the course.