CSCI 301: Formal Languages and Functional Programming
Syllabus, Fall, 2018

Instructor: Geoffrey Matthews
Email: `geoffrey dot matthews at wwu dot edu`
Office hours: MTWF 9:00-9:50, CF 469

TA: Evan Ricks, `rickse2 at wwu dot edu`

Meeting times:

CRN	Lecture			Lab		
41773	8:00-8:50	MTWF	CF225	M	CF164	12:00-1:50
41774	8:00-8:50	MTWF	CF225	W	CF164	12:00-1:50

You may attend a lab session you did not sign up for on a space available basis (you may not bump someone who has registered for that section). There are no labs the first week of class.

Learning environment: I am committed to establishing and maintaining a classroom climate that is inclusive and respectful for all students. Learning includes being able to voice a variety of perspectives, and classroom discussion is encouraged. While expressed ideas may vary or be opposed to one another, it is important for all of us to listen and engage respectfully with each other.

Catalog copy: Introduction to discrete structures important to computer science, including sets, trees, functions, and relations. Proof techniques. Introduction to the formal language classes and their machines, including regular languages and finite automata, context free languages and pushdown automata. Turing machines and computability will be introduced. Programming using a functional language is required in the implementation of concepts. Includes lab.

Goals: On completion of this course, students will demonstrate:

1. Thorough understanding of the mathematical definitions of concepts important to computer science, including sets, tuples, lists, strings, languages, graphs, trees, functions and relations
2. The ability to prove basic theorems involving these mathematical concepts
3. The ability to employ effectively the functional programming style in a functional programming language
4. Solid understanding of fundamental classes of languages, including regular and context free, and their corresponding machines
5. Basic understanding of Turing machines and computability
6. Basic understanding of important algorithms, including conversion of finite automata to different forms, conversion of grammars to machines
7. Basic understanding of LL(k) and LR(K) grammars and the parsing techniques used for those grammars

Websites:

- For class materials: `https://github.com/geofmatthews/csci301`
- For turning in homework and grading: `https://wwu.instructure.com/`

Goals: This class is an introduction to computer science *theory*. This is exactly parallel to the distinction between theoretical physics and applied physics. We will use simplified, abstract, ideal mathematical models of computers, so that we can study the theoretical limits of what they can accomplish. We will begin studying the basic mathematics we need, and then study mathematical models of computers of increasing complexity and power. Two classes of computers, finite state automata and push-down automata, also form the basis of powerful programming paradigms useful in a variety of situations.

We will also study the functional programming language *Scheme*. Its simplicity, power, and mathematical elegance will inform our study of computers in the abstract, and also teach us new styles of programming.

Math texts:

Required:

- `http://www.people.vcu.edu/~rhammack/BookOfProof/`
- `http://cg.scs.carleton.ca/~michiel/TheoryOfComputation/`

Optional:

- `http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-045j-automata-computability-and-complexity-spring-2011/lecture-notes/`
- `https://www.tutorialspoint.com/automata_theory/`
- `https://www.geeksforgeeks.org/theory-of-computation-automata-tutorials/`
- `http://users.utu.fi/jkari/automata/`
- `http://maths.mq.edu.au/~chris/notes/second_langmach.html`

Programming texts:

Required:

- `http://ds26gte.github.io/tyscheme/`
- `http://www.scheme.com/tspl3/` Note: Use the 3rd edition and do *not* use the 4th edition of this book. Our software is based on R5RS scheme and the 3rd edition covers this. The 4th edition is based on R6RS scheme.
- The *help desk* within Racket leads to documentation on the implementation.

Optional:

- `http://mitpress.mit.edu/sicp/`
- `http://www.shido.info/lisp/idx_scm_e.html`
- `https://classes.soe.ucsc.edu/cmps112/Spring03/languages/scheme/SchemeTutorialA.html`

Software: *DrRacket*, available here: `http://racket-lang.org/`

Quizzes: Pop quizzes may be handed out at any time. After working on them individually we will solve them together in class. They must be turned in during class for credit. No makeups.

Exams: A midterm and a final, according to the schedule below. The final will be cumulative.

Exams are closed book, with the exception that you may consult two pieces of paper during the exam. You may write or print whatever you wish on both sides.

Laboratory exercises: Laboratory exercises will be handed out every week except the first and last weeks of class and Thanksgiving week (no classes Wednesdays). They will involve programming in *Scheme*. Each lab is due by midnight Monday of the following week. Note that the last lab will be due Monday of dead week. No late work will be accepted.

Homework problems: Math assignments must be formatted using the LaTeX technical typesetting system and submitted online. Turn in both the `.tex` file and the `.pdf` file. Do not zip or combine them into one file.

Points will be awarded to each assignment based on its difficulty. Note that there may be an assignment due during dead week.

Ample time to complete the assignments will be allowed. Late work will not be accepted.

Pedagogically, it is important that you do as much independent work as possible. The assigned homework represents a minimum, but there are many exercises in the books and online to get more practice at the math involved.

We will attempt to grade the homework in a timely fashion; this may necessitate grading only a sample of the problems submitted.

Assessment and grades: Your grade will be based on the math homework, labs, quizzes, and the two exams. Weights are as follows.

Grades will be assigned based on scores as shown. At the discretion of the instructor, scores may be scaled. Awarding ± is also at the discretion of the instructor.

Homework	Labs	Quizzes	Midterm	Final
25%	25%	5%	15%	30%

%	90-100	80-89	70-79	60-69	0-60
Grade	A	B	C	D	F

Schedule:

Week		Readings *TYS*	*TSPL*	*BoP*	*ITC*	Labs	Notes
Sep	24	1	1	I			
Oct	1	2,3,4	2	II & III		Lab 1	
Oct	8	5,6,7	3	IV	1	Lab 2	
Oct	15				2	Lab 3	
Oct	22				2	Lab 4	
Oct	29				3	Lab 5	
Nov	5				3	Lab 6	Midterm Friday, Nov 9
Nov	12				4	Lab 7	
Nov	19				5	Wed Lab 8	Holiday Monday
Nov	26					Mon LaB 8	Holiday Wed-Fri
Dec	3			Review		Lab 8 due Dec 3	
Dec	10	Final Exam, Mon, Dec 10, 1:00-3:00pm					

Book	URL
TYS	`http://ds26gte.github.io/tyscheme/`
TSPL	`http://www.scheme.com/tspl3/`
BoP	`http://www.people.vcu.edu/~rhammack/BookOfProof/`
ITC	`http://cg.scs.carleton.ca/~michiel/TheoryOfComputation/`

Attendance policy: Attendance is not required but strongly recommended. Studies show that regular attendance is highly correlated with performance.

You must be in attendance to receive credit for a pop quiz. There are no makeups for pop quizzes.

You are responsible for all material covered in the lectures, books, handouts, or other assigned reading. If you miss a lecture, make sure you get notes from another student.

If you have an emergency that necessitates your absence from class, notify me as soon as possible, preferable before the absence. I will handle each case individually and may, for example, extend the due date for the assignment, schedule a make-up exam, or simply adjust your remaining scores to determine your grade.

Academic dishonesty: Please read Appendix D of WWU's Catalog on Academic Dishonesty. It is available online at `http://catalog.wwu.edu`.

Unless specified otherwise, all work for this course is meant to be done **individually.** The work that you turn in for a grade must be completely your own, or you will be guilty of academic dishonesty and could receive an F for the course.

However, it can be a valiable learning experience to discuss work with your fellow students, and this is encouraged. However, after working with a colleague, **you may not keep any paper or electronic copies of anything you produced together!** You may only keep your memories. In particular, this means that **you may not ask for or give help while sitting in front of a computer where the assignment is open!** Also, **you may not use anything a colleague has emailed to you!** Delete the email and do not save a copy.

To help understand what I mean, remember the **Long Term Memory Rule**. You may discuss, sketch, write things down, use your computers, whatever, but after you are done working with your fellow students all files must be deleted, whiteboards erased, and all papers you created must be destroyed. You should then watch a rerun of *the Simpson's*, play a game of ping-pong, take a walk, or something else for half an hour. After this you can go back to your assignment (alone) and use the knowledge you have now gained.

It is very easy for experienced software developers like your instructor and your TA to detect copied assignments. Please do not put us in a situation where we have to fail you for plagiarism.