CS 241E: Foundations of Sequential Programs (Enriched)
Fall 2018

Instructor: Ondřej Lhoták (olhotak@uwaterloo.ca, DC 2520, x36654)
ISA: Daniel Zheng (cs241e@uwaterloo.ca, MC 4065)
Instructional Support Coordinator: Gang Lu (glu@uwaterloo.ca, MC 4008, x38243)
Lectures: 11:30 – 12:50pm TTh in PHY 313
Tutorials: 4:30 – 5:20pm W in PHY 313 starting September 12th
Objective: To describe the relationship between high-level programming languages and the computer architecture that underlies their implementation.

Prerequisites: (85% in CS 146) or (85% in CS 136) or (85% in CS 138); Computer Science students only.
Antirequisites: CS 230, GENE 344
Non-enriched version: Any CS 241E student may switch to the regular version of CS 241 during the **first two weeks** of lectures. Please see the CS undergraduate advisors to do so.
Website: http://www.student.cs.uwaterloo.ca/~cs241e. Essential announcements and assignment specifications can be found here.
Discussion Forum: https://piazza.com/uwaterloo.ca/fall2018/cs241e. Essential announcements will be posted here and it is a handy and timely way for students to learn from each other and the course staff.
Course Description: The relationship between high-level languages and the computer architecture that underlies their implementation, including basic machine architecture, assemblers, specification and translation of programming languages, linkers and loaders, block-structured languages, parameter passing mechanisms, and comparison of programming languages.
Course Overview:
Machine architecture and assembly language (6 hours)
Functional components of a computer: memory, control unit, arithmetic/logic unit, input/output devices. Data representation. Machine language: operation codes, addressing modes, indexing, base registers, register designation. Assemblers. Linking and Loading.

Machine-level implementation of high-level language features (9 hours)
Variables and data representation, control structures, procedures and procedure calls, scopes and nesting, closures and objects.

Regular languages and scanning (5 hours)
Architecture of a compiler. Syntax vs. semantics. Introduction to formal languages. Regular languages, regular expressions and finite state machines.

Context-free languages and parsing (5 hours)
Context-free grammars, derivations, derivation trees, ambiguous grammars. Parsing algorithms.

Semantic Analysis and Code generation (6 hours)
Parse trees, name resolution, type-checking, code generation.

Memory Management (5 hours)
Implications of stack versus heap allocation. Dynamic Memory Allocation. Automatic Garbage Collection.

Assignments: There will be eleven assignments. Real-time feedback on the correctness of your solutions is provided by the Marmoset submission and testing server. All assignments must be submitted electronically to Marmoset and results are normally quickly available. We recommend that you work on assignments early. Use Marmoset to assess your progress (and grade!) after convincing yourself of correctness using your own self-designed tests. By the time you submit to Marmoset, you should be convinced by your own thorough testing that your program is perfect. A link to the Marmoset system and instructions for using it may be found on the course web page.

To receive full credit, a submission *must* be received before the due date/time of the assignment. A

submission received up to one week late will receive partial credit calculated as 75% of the grade on the late submission plus 25% of the grade on your best on-time submission for the same assignment.
Midterm Exam: Wednesday, Oct 24th, 2018 from 7:00pm to 8:50pm. Rooms will be announced later.
Final Exam: There will be a $2\frac{1}{2}$-hour final exam held during the examination period.
Grade Calculation: Assignments 25%, Midterm Exam 25%, Final Exam 50%.
You must achieve 50% or more on the weighted average of the midterm and final exams in order to pass the course. If you do not meet this minimum requirement then your final grade for the course will be at most this weighted average of the midterm and final grade.

Academic Offenses: Students are expected to know what constitutes academic integrity, to avoid committing academic offenses, and to take responsibility for their actions. Students who are unsure whether an action constitutes an offense, or who need help in learning how to avoid offenses (e.g., plagiarism, cheating) or about rules for group work/collaboration should seek guidance from the instructor, ISC, ISA, or the Undergraduate Associate Dean. For information on categories of offenses and types of penalties, students should refer to Policy #71, Student Academic Discipline,
http://www.adm.uwaterloo.ca/infosec/Policies/policy71.htm. Students who believe that they have been wrongfully or unjustly penalized have the right to grieve; refer to Policy #70, Student Grievance,
http://www.adm.uwaterloo.ca/infosec/Policies/policy70.htm.

All assignments in CS241E are to be done individually. You are welcome to discuss general ideas regarding assignments with other students in the class, but no code-level sharing is permitted. You may not look at someone else's code, nor share your code with someone else, either in person or via electronic communication (e.g., instant messaging, discussion forum). When code is shared, **both** parties are considered to have committed an offence. Marmoset tokens cannot be shared; it is an offence to "borrow" someone else's Marmoset account for the purpose of using extra release tokens for testing, or for any other purpose.
It is an offence to submit for credit anything that has previously been submitted for credit in the same or any other course, unless permission is explicitly granted to do so. Although each assignment is worth only about 2% of your final grade, the penalty for an offence under Policy 71 is a grade of 0 on the assignment and an *additional* 5% deduction from your course grade.
Use of MOSS: MOSS (Measure of Software Similarities) is used in CS 241E as a means of comparing students' assignments in order to support academic integrity.

Note for students with disabilities: AccessAbility Services (AAS), located in Needles Hall 1132, collaborates with all academic departments to arrange appropriate accommodations for students with disabilities without compromising the academic integrity of the curriculum. If you require academic accommodations to lessen the impact of your disability, please register with the AAS at the beginning of each academic term.

Course Schedule: A course outline follows. Note that the assignment of topics to weeks is our current best guess as to what will be covered when. If you miss a lecture, it is your responsibility to get the material you missed from a friend or nearby classmate. Assignment due dates are subject to change as dictated by lecture pace.

Week	**Lecture Dates**	**Topics**	**Notes**
I	Sept 6, 11	Introduction, machine language	Tutorials begin
II	Sept 13, 18	Assembly language, assemblers	Assignment 1 due M Sept. 17
III	Sept 20, 25	Variables, memory allocation	Assignment 2 due F Sept. 21
IV	Sept 27, Oct 2	Control structures	Assignment 3 due F Sept. 28
V	Oct 4, 11	Procedures and calls	Assignment 4 due F Oct. 5
VI	Oct 16, 18	Scopes, closures, objects	Assignment 5 due W Oct. 17
VII	Oct 23, 25	Regular languages, scanning	Midterm W Oct. 24
VIII	Oct 30, Nov 1	Context-free languages, parsing	Assignment 7 due W Oct. 31
IX	Nov 6, 8	Context-sensitive analysis	Assignment 6 due W Nov. 7
X	Nov 13, 15	Code generation	Assignment 8 due M Nov. 12
XI	Nov 20, 22	Memory management	A9 due M Nov. 19, A10 due F Nov. 23
XII	Nov 27, Nov 29	Memory management	Assignment 11 due M Dec. 3