605.615.8VL Spring 2022
Compiler Design with LLVM
Course Outline

This outline provides an overview of the course and assignments by Class (**subject to change**).

Reading	Topic	Class & Date	Programming Assignments	Spreadsheet Front End & JIT (35%)	Compiler Project Due (30%)
See **Blackboard**: *Preparation Modules* Module 0: Preparation *Using Clang … Data Files*	LLVM Environment	Prep	Download LLVM executables and source **Due W1**		
Read **Louden** Chapters **1.0–1.5 (Compiler Organization)**, and **2.0-2.3.0 (Scanning)**	Introduction & Overview Fundamental Definitions Compiler Organization Chomsky Hierarchy Intro to LLVM ASCII	W1 – 1/27 Quiz1	Standalone Scanner – **Due W2**		Download LLVM executables and source
Read **Louden** Chapter **3 through 3.6.3 (skip 3.5.2) (Parsing) and 4.0 through 4.1.3 (Recursive Descent) and 4.4 (Appendix B for reference)**	Scanning Intro to Lexical Analysis Finite State Machine Regular Expressions Example Scanner	W2 – 2/3 Quiz2	Standalone Scanner with **FSM – Due W3** SS – Add Scanner to Skeleton SS– **Due W3**	Standalone Scanner **(2%)**	
Read **Louden** Chapter **4.2–4.3.4 (LL(1)), 4.5-4.5.2 (Error Recovery) and 5.0-5.2.1, 5.3.1–5.3.2, 5.4, 5.7.1-5.7.2(LR) Read about ANTLR4 and Def. ANTLR4 Ref. 2.0-2.2**	Parsing Parse Tree and AST BNF and EBNF LL LR & RD Parse Demos Left Recursion Recursive Decent Parsing ANTLR4 Parsing	W3 – 2/10 Quiz3	SS – Add Recursive Descent Parser, make AST **Due W5** CP Make ANTLR4 C– Scanner / Parser **Due W4**	Standalone Scanner **with FSM (2%)** SS – Add Scanner to Skeleton SS **(5%)**	

Read about **ANTLR4** C++ Runtime. Read **Louden** Chapter **6.0, 6.1.0 and 6.2.2–6.3.0** Read **Kaleidoscope Tutorial Ch. 1-3 (to IR Gen)**	Parsing (Continued) AST gen RD Parsing (Cont.) First and Follow sets LL(1) Parsing Error Recovery	W4 – 2/17 Quiz4	SS – Add Semantic Error Handling – **Due W6** CP – ANTLR4 Semantic Analysis in C++, **Due W6**		CP – (java) Cminus scanner / Parser **(3%)**
Read **Louden Chapter 6.3.2–6.3.4 and 6.4–6.4.5** Read **Kaleidoscope Tutorial Ch. 4 (JIT) Skip "Trivial Constant Folding" and "LLVM Optimization Passes"**	Parsing (Continued) LR Parsing intro LR Parsing Errors Semantic Analysis Symbol Tables LLVM IR generation Intro	W5 – 2/24 Quiz5	SS - IR Generation– **Due W8** CP – IR Generation– **Due W8, W9 & W10**	SS – Add Recursive Descent Parser and Produce AST **(6%)**	
Read **Louden** Sect. "**The Source Code Optimizer**" on **pages 10-11** Read **Ch. 8.0-8.1.0** (LLVM IR is Three-Address code) and **Ch. 8.9.0–8.9.2** Read **Kaleidoscope Tutorial Ch. 4 (Optimization)and Ch. 5 (Functions and Control Flow)**	LLVM IR generation (cont) IR organization, Simple IR for Function calls Intro Optimize Passes Precompiler, Review for Midterm	W6 – 3/3 Quiz6	SS add & call JIT – **Due W9** Research – Opt. Pass Research– **Due W8, W9 & W10**	SS – Add Semantic Error Handling **(5%)**	CP Semantic Analysis in C++ **(4%)**
No New Reading Assignment	**Midterm** **(15%)** in class	W7 – 3/10 & Midterm	No new Homework		

Read **first five** sections of "**Writing an LLVM Pass**" ***** and "**Pass registration**" through "**The release Memory method**."	Midterm Recap SSA & Phi Functions, Optimizations. Linking & Code Gen Dynamic Link & JIT	W8 – 3/17	CP – write Opt. Pass – **Due W11**	SS - IR Generation with Print Out **(5%)**	CP – IR Generation of equation **(3%)** Readable IR Examples of Opt. Pass Research Before and After **(1%)**
	Spring Break	SB – 3/24			
Read **Kaleidoscope Tutorial Ch. 6 & 7**	Optimization (cont.) Analysis passes, Pass Manager	W9 – 3/31 Quiz9	SS–add function call **Due W12**	SS add & call JIT **(5%)**	CP – IR Generation of "if" test **(3%)** Paper on Opt Pass Research **(1%)**
Read **Louden** Ch. **7.0 – 7.2** (Runtime Environments) Read remainder of the **Kaleidoscope Tutorial**	**Present Opt Research**	W10 – 4/7 Quiz10			CP – IR Generation and Print Out **(3%)** Opt Research Presentation **(6%)**
Read "**Beginner's Guide to Linkers**" skip "Windows DLLs" through "Templates" Read "**LLVM Link Time Optimization**"	Target Code Generation Runtime Environments Miscellaneous Topics	W11 – 4/14 Quiz11	CP – Target Code Gen – Link & Run – **Due W12**		CP Opt pass **(3%)**
	Linkage Editor Loader	W12 – 4/21 Quiz12	CP – Virtual Machine Mods – **Due W13**	SS–add function call **(5%)**	CP – Target Code Gen – Link & Run **(2%)**
	Cross compilation Review for Final	W13 – 4/28	**All Homework Due**		CP – Virtual Machine Mods **(1%)**
	Final **(10%)** in class	W14 – 5/5			

Class Participation and Quizzes (10%)

*** SCC** = **Strongly Connected Component** is a group of nodes within a directed graph in which any node can be reached from any other node within the SCC group (like the body of a loop or a recursive descent parser for a sequence of statements).

Region a group of basic blocks that have a single entry point and a single exit point (like a pure function or an “if” statement). Regions can be nested and should be processed inner most first.

Dominator a node in a graph through which control must pass to get to a specific node. The immediate dominator is always unique. Therefore the dominator graph is always a tree.

Critical Edge a point in the control flow of basic blocks where an edge from a block with multiple successors connects to a block with multiple predecessors.