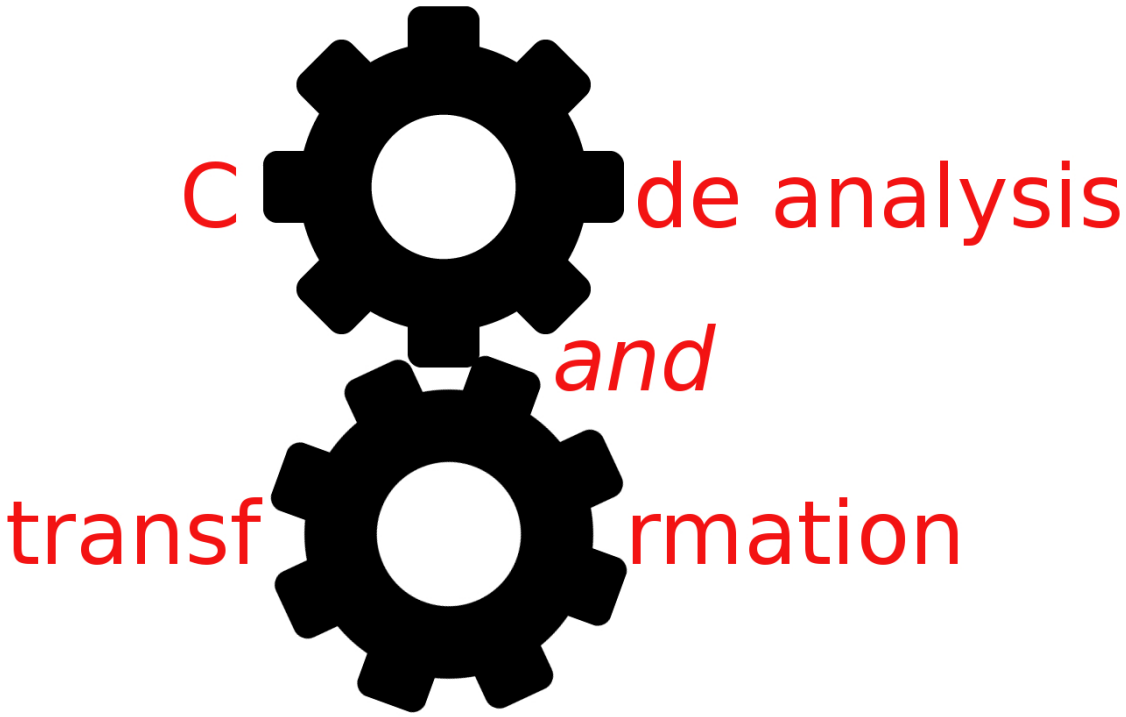

Tentative Schedule

Legend

C = Chapter of the class book
P = Set of papers
H = Homework
M = Material archive

Time schedule					
Week	**Date**	**Topic**	**Reading, Materials**	**Homework**	**Homework due**
1	9/22 9/24	Welcome to Code Analysis and Transformation! Introduction to LLVM	C 1 M 1, P 1	 H 0	 9/30
2	9/29 10/1	Makefile, LLVM A few examples of LLVM passes, Control flow analysis	M 2, P 1 P 1, C 8.4 (w/o C 8.4.2), M 3	 H 1	 10/11
3	10/13 10/15	Intermediate representation (SSA), Code optimizations Introduction to data-flow analysis, reaching definition analysis	C 6 - C 6.2, C 9.1, M 4 C 9.2, M 5	 H 2	 10/21
4	10/20 10/22	Constant/copy propagation, deadcode elimination, CSE Foundations of data-flow analysis	C 9.1 - 9.2, C 9.4, M 6 C 9.3	 H 3	 10/28
5	10/27 10/29	Loops and their identification Loop normalizations in LLVM	C 9.6, M 7 M 8	 H 4	 11/4
6	11/3 11/5	Loop transformations Dependence analyses	P 2 P 3	 H 5	 11/18
7	11/10 11/12	Dependence analysis in LLVM Interprocedural code analysis	 C 12	 H 6	 12/2
8	11/19 11/24	Local code scheduling, ILP, and data locality Global code scheduling	C 10-10.3 C 10.4	 	
9	12/1 12/3	Partial-redundancy elimination Traces and superblocks	C 9.5 P 4	 	
10		Your whole-project presentation	Your paper		

Papers	
P1	LLVM website LLVM documentation
P2	Sections 6.1, 6.2, 6.3, 6.4 of Compiler transformations for high-performance computing
P3	The program dependence graph and its use in optimization
P4	The superblock: an effective technique for VLIW and superscalar compilation

Code archives	
M 1	llvm.tar.bz2
M 2	llvm_2.tar.bz2
M 3	llvm_3.tar.bz2
M 4	llvm_4.tar.bz2
M 5	llvm_5.tar.bz2
M 6	llvm_6.tar.bz2
M 7	llvm_7.tar.bz2
M 8	llvm_8.tar.bz2

Homeworks	
H 0	Print functions compiled
H 1	Print the number of static invocations of CAL functions
H 2	Print GEN and KILL sets for reaching definitions
H 3	Print IN and OUT sets for reaching definitions
H 4	Global constant propagation with assumptions in the input code