Course Summary

“The stone it called to me,
(and now I see the things the stone has shown to me)"

They Might Be Giants
The Statue Got Me High

Parsing
- Lexing vs. Parsing
- Recursive Descent / LL(1)
- Shift-Reduce / LR(1)
- PEG / Packrat Parsing
- Lex, Yacc, ...

Compiler Targets
- x86 architecture
- RISC architectures
- LLVM
- JVM

Low-level issues
- Parameter passing (cdecl, etc.)
- Tailcalls
- Data representations (data as bits)
- Register allocation (graph coloring, optimism, coalescing)
- Code generation (dynamic programming, maximal munch)
- Instruction Scheduling (list scheduling)

Program Analyses / Optimization
- Decidability
- Dataflow theory (lattices, fixed points)
- Liveness analysis
- Redundancy elimination (value Numbering, CSE, PRE)
- Dead code elimination
- Loop optimizations (invariant hoisting, induction variable elimination)
- Inlining
- Induction variable elimination
- Strength reduction
- Constant folding, constant propagation, copy propagation

Intermediate Languages
- Three-address code / quadruples
- Static Single Assignment (SSA) form
- Linear vs. Tree vs. control flow graphs

Type Checking
- Type Checking
- Subtyping (implicit conversions)
- Overloading
- Type inference

Memory management
- Static / stack / heap allocation
- Activation records
- Garbage collection (reference counting, mark-sweep, copying, generational)

Source languages
- Imperative languages (control flow)
- Functional/higher-order languages (closures, pattern matching, ...)
- Object-oriented languages (virtual method tables, ...)

Other Tools
- Profilers
- Partial evaluators
Postscript: Some Recommended Further Reading

- Steven S. Muchnick, *Advanced Compiler Design and Implementation*. A very good text if you're interested in compiler optimizations.

- Henry S. Warren, Jr., *Hacker's Delight*. Filled with all sorts of cool bit-level tricks (e.g., reordering and rearranging and counting bits and bytes, computing signed arithmetic and comparisons with unsigned operations and vice-versa, multiplying by a constant via shifts and adds, division by a constant via multiplication, etc.) Chapter 2 (“Basics”) is available on-line at [http://www.hackersdelight.org/](http://www.hackersdelight.org/).

- Stanley B. Lippman, *Inside the C++ Object Model*. A very readable description of how C++ compilers implement constructors, inheritance, multiple inheritance, virtual inheritance, and so on.


- There's a lot more to LLVM than what we covered in class: [http://llvm.org](http://llvm.org)

- Further references are available on the course web page.