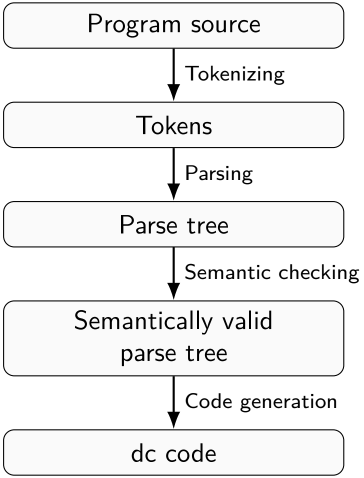
CSC-363 Lecture 04C
Code Generation

Colin McKinney

13 February 2026

Code Generation

We're ready for the last stage of a complete compiler: code generation. Our process:



Code Generation

- We could theoretically generate to an intermediate representation and then transform that to code. We won't here, but this can be useful in "real" compilers to add multiarchitecture capability or do various types of optimization.
- We'll instead go directly to dc code, so we can actually execute it using dc.
- Code generation will look somewhat like semantic checking, in that we will need to traverse each AST recursively and build up code from leaf to root.
- We won't have to worry about some things, like register allocation, because we're treating the dc registers as being synonymous with variables AND we're limiting the valid variable names.
- In a "real" program, we'd be able to have as many variable names as we wanted, and should be able to get away with as few as 1-4 registers assuming we have a stack structure to work with. More on this when we get to microC/RISC-V.