Course Outline and Syllabus
Compilers : Theory and Practice (3 credit hours)

- Instructor Contact Information
  - Prof. Santosh Pande (E-mail: santosh@cc.gatech.edu)
  - Office : KACB 2338

- Learning objectives:
  - To learn the theory and practice behind various phases of automatic translators (compilers) for higher level programming languages such as : scanners, parsers, semantic analysis, code generation, register allocation, and instruction selection.
  - To engineer and build key phases of a compiler in Java for a mini language.

Pre-requisites and expected background: Solid knowledge of data structures and algorithms, very good programming skills and ability to write sizeable software in Java. Should be able to work in Linux environment. Must have done a basic course in computer architecture

Topics

- Front End: Compiler phases and overall working, Scanners (Regular Expressions, NFA/DFA, Scanner Generators), Parsers (Grammars, Ambiguity Removal, LL, LR and other deterministic parsing), Semantic Analysis (Symbol Tables, Syntax Driven Analysis, Type Systems, Attribute Grammars)
- Middle End: Intermediate Forms, Syntax Directed Translation, Translation of: Expressions, Control Structures and Back-patching, Function and Procedure Calls, Runtime Organization
- Back-end: Instruction Selection, Register Allocation, Instruction Selection, Code layout and Code Generation, , Brief Introduction to Code Optimizations

Course Material:

- **Required textbook:**
  Engineering A Compiler (2nd Edition)
  Authors: Keith D. Cooper and Linda Torczon
  Morgan Kaufmann Publishers 2nd Edition (February 21 2011)
  ISBN 10**:** 012088478X
  ISBN 13**:** 978-0120884780
- **Optional Supplement :** Course slides and handouts: To be posted on Canvas

Workload and grading (tentative)

- Home-works: About 4-5 exercises (some of them could involve programming, some just paper and pencil exercises): 30% weightage
- Mini-projects: A series of projects to build a working compiler in a phased manner – spread out throughout the semester 35% weightage
- Finals: 35% weightage (comprehensive covering all the topics)

Course Educational Outcomes:

Upon successful completion of this course, students should be able to …

- Devise and implement finite state machines (deterministic automatons) that recognize legal words made out of the alphabet of a language as per the underling language's lexical rules.
- Devise and implement predictive parsers that facilitate syntactic checking of input programs as per the grammar of the underlying language.
- Devise attribute grammars and implement semantic checkers that determine soundness of programs as per the underlying type system and scoping rules specification.
- Devise safe code generation techniques as per language's definition.
- Devise optimizations for register allocation and instruction selection to improve code efficiency.
- Devise an intermediate representation which will be used during the translation. Design the flow and transformations of the value attributes to be maintained during the translation.

Academic Integrity

Academic dishonesty will not be tolerated. This includes cheating, lying about course matters, plagiarism, or helping others commit a violation of the Honor Code. Plagiarism includes reproducing the words of others without both the use of quotation marks and citation. Students are reminded of the obligations and expectations associated with the Georgia Tech Academic Honor Code and Student Code of Conduct, available online at www.honor.gatech.edu. All the software turned in as a part of this course will be thoroughly checked for plagiarism and fullest penalties will be imposed upon detection of violations of honor code.

Learning Accommodations

If needed, we will make classroom accommodations for students with documented disabilities. These accommodations must be arranged in advance and in accordance with the Office of Disability Services (http://disabilityservices.gatech.edu).

Excused Absence Policy http://www.catalog.gatech.edu/rules/4/