# CSC 7351
# Advanced Compiler Design Theory
# Fall 2018

# Syllabus

Gerald Baumgartner

## Course Summary

Automatic generation of LL(1), LR(1), LALR(1) parsers, syntax directed translation of high-level control structures, error recovery, optimization of branching, local code optimization using directed acyclic graphs, loop optimization, global data flow analysis and object-code optimization.

## Prerequisite

According to the course listing, the prerequisites are:

> CSC 4351 or equivalent.

## Office Hours

| Who | Where | Phone | E-Mail | When |
|---|---|---|---|---|
| Gerald Baumgartner | Patrick F. Taylor Hall 3272D | 578–2191 | gb | MW 2:30–4:00am |

Other office hours by appointment (recommended).

## Important Dates

- Labor Day: Mon, Sep 3
- Midterm: Wed, Oct 10
- Turkey Day, Wed, Nov 21
- (Final: Mon, Dec 3, 5:30–7:30pm)

All exams are comprehensive.

## Reading

- Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman, Monica S. Lam, *Compilers: Principles, Techniques, and Tools*. Addison-Wesley, 2006 (optional).
- Keith D. Cooper, Linda Torcson, *Engineering a Compiler*. Morgan Kaufmann, 2011 (optional).
- Andrew W. Appel, *Modern Compiler Implementation in Java*. Cambridge University Press, 1998 or 2002 (optional).
- Thomas W. Parsons, *Introduction to Compiler Construction*. Computer Science Press, 1992 (optional).
- Des Watson, *A Practical Approach to Compiler Construction*. Springer Verlag, 2017 (optional)
- `http://docs.oracle.com/javase/tutorial/`
- `http://docs.oracle.com/javase/specs/`
- `http://docs.oracle.com/javase/8/docs/api/`
- Course web page `http://www.csc.lsu.edu/~gb/csc7351/` and Moodle

## Projects

There will be one or two programming projects of varying difficulty. The scope and duration of the programming assignments is yet to be determined. The projects can be implemented in groups of one or two.

A penalty of 10% will be assessed for each day a project is late up to a maximum of 30% after which the project will not be accepted. The final project may not be turned in later than Saturday midnight after the last day of classes.

## Homeworks

There will be short homework assignments. The homeworks will not be graded. They are purely for practice.

## Presentation

You will be asked to give a short presentation on a compiler architecture topic.

## Annotated Bibliography

There will be an annotated bibliography and short paper on a compiler topic of your choice.

## Grading

| | |
|---|---|
| Projects | 40% |
| Midterm | 20% |
| Presentation | 10% |
| Bibliography | 10% |
| Final | 20% |

Both exams are comprehensive.

The final exam will be an oral exam consisting of a demo of the programming project and questions about the project and other topics from the course. If the project was implemented in a group of two the oral exam will be with both students at once.

## Topics

The exact list of topics is yet to be determined. We will cover lexical analysis and parsing, with some theory but an emphasis on tools, semantic analysis and compiler architecture, and optimizations (data flow analysis, model-driven search-based optimizations, and polyhedral optimizations). The topics will be adjusted to fit with what we need for the programming assignments. We have flexibility in the course to adjust the topics based on the students' interests.

## Due Dates and Grading

Since the time needed for finishing a programming assignment is hard to estimate and to allow fixing severe bugs that show up close to the deadline, programming assignments can be submitted up to three days after the official deadline. The last programming assignment cannot be submitted later than Saturday midnight after the last day of classes. For each day past the deadline, a penalty of 10 percent will be incurred. Programming assignments will be submitted electronically. They will be due at midnight.

## Programming Standards

The algorithm used must be essentially correct. Obviously, the program should compile and run. Because of the complexity of some of the programs, no credit can be given for a program that doesn't run. If a program dumps core (throws a run-time exception), only partial credit will be given.

Since projects build on top of previous projects, it is very important to get each submission to run without core dumps and to structure the program so it can be easily extended.

I expect your work to exhibit high standards of programming style and layout, reflecting your expertise as a computer professional. Poor style and documentation may results in up to 10% deducted.

## Honesty

I will treat you as professionals, and you should plan on conducting yourself as such. This course presents many important concepts you will need throughout your career as a computing professional, so it is important that *each student* do *all* the assignments and projects and learn the material.

There will be several homework assignments and programming projects. You are free to discuss these assignments with others. However, the programs and homework solutions you submit are to be developed by yourself. *Cheating is a very serious offense and will not be tolerated*. I may use tools for detecting cheating on programming assignments. Copying code from the internet is also considered cheating, except for small code fragments if they are attributed appropriately and if using the code is permitted by the author's copyright. Supplying others with material is also against this rule. The policy is that the supplier and receiver of information will both be reported to the Dean of Students.

Save all handwritten notes and printouts you generate as you work on a project and keep them until the end of the quarter so as to protect yourself in the event that someone "borrows" your program, or the version you submit is mislaid. For your protection, cases of missing output should be *immediately* reported.

## Computer Account Security and Use

To help others resist the temptation of using your work, you should maintain proper security on your computer account. Especially, keep your password from others and do not alter the protection on any of your files. To give others access to your account or files or printouts of your programs is the same as giving them the information directly and will be dealt with accordingly. Any trouble with computer accounts should be referred to an instructor as soon as possible.

When a program has been submitted electronically, you should maintain an *unedited* version of what you submitted (with the correct date stamp) until after that program has been graded. It is also beneficial to use version control software such as git or CVS to keep track of all versions of files so you can revert back to an old version if necessary.