

# CompSci 654/754: Compilers

John Boyland

compsci-654@uwm.edu

Spring 2012

	John Boyland	Chao Sun
<b>Office</b>	EMS 925	EMS E280
<b>Office Hours</b>	T 2:00–3:00pm, R 10:00am–12:00pm, F 8:30–9:30am	M 4:00–6:00pm
<b>Phone Number</b>	229-6986, 964-3227 (home)	

A compiler is a program that translates another program from one language to another. The first language is usually a high-level language such as C++, Java or Common Lisp. The second language is often a low-level language such as MIPS assembly language or Java byte code. As part of the course, students will construct a working compiler for an object-oriented language, and then use it to execute real programs in the language.

## 1 Desired Outcomes

Students will

- know the basic structure and functionality of compilers
- be able to apply formal language theory to construct compiler front-ends
- understand symbol tables
- know how high-level features such as objects and method calls are implemented
- be familiar with basic optimizations, including redundancy elimination, copy propagation, and dead-code elimination
- be familiar with issues of register allocation and assignment
- be able to construct a working simple compiler for a simple object-oriented language to a simple but realistic machine language using syntax-directed translation

## 2 Requirements

Junior standing, CompSci 417, and CompSci 431 are prerequisites.

Students must be

1. able to write, compile and execute programs in a UNIX environment
2. able to use makefiles
3. able to define classes with information hiding (abstract data types ADTs)
4. able to use and implement standard data structures (dynamic arrays, linked lists, binary search trees and hash tables)
5. able to use inheritance and virtual member functions to implement designs using dynamic-dispatch for polymorphism
6. able to understand and modify programs with tens of classes

7. able to write black-box unit tests of simple ADTs
8. able to understand and write regular expressions and context-free grammars
9. familiar with functional programming

### 3 Texts

The required textbook for the course is

Michael Scott. *Programming Language Pragmatics*. Morgan Kaufmann, San Francisco. 3rd edition. 2009. (Earlier editions are also acceptable, but the chapters and page numbers come from the third edition.)

Assigned readings are given at the end of this syllabus.

I also recommend that one read the classic compilers textbook:

Alfred V. Aho, Ravi Sethi and Jeffrey D. Ullman. *Compilers: Principles, Techniques and Tools*. Addison-Wesley, Reading, Massachusetts.

This book has come out in a third edition.

### 4 Discussion Section

The discussion section will be used to present specific information about the tools we are using (for example, coollex, coolyacc and spim) as well as on the structure of the many files that form the infrastructure of the Cool compiler. There will also be time for concrete questions on the programming assignments.

### 5 Grading

The grade for the course will be computed from the following parts:

#### 30% Examinations

There will be three examinations spaced throughout the course; each examination contributes 10% to the course grade. Any single examination may be made-up during the final examination time.

#### 42% Programming Assignments

There will be eight programming assignments, each worth 6% of the course grade. The lowest assignment grade will be dropped.

#### 18% Homework

There will be a written homework every second week (7 in all). Each homework is worth 3% of the grade. The lowest grade will be dropped.

#### 10% Reading

Over the course of the semester, there will be 12 quizzes on the required reading. Each counts 1% of the course grade and the lowest two will be dropped.

Late assignments will not be accepted.

Exceptions to these rules can be made only in extraordinary circumstances. Advance notice of a need for an exception should be given if possible. All graded assignments must be your own work (your own words), but you may work with other people as long as you list their names prominently on the first page of the turned in assignment or homework. Unless specific permission is given (for example, in group assignments), no electronic or xerographic copying of assignments is permitted. Whether or not you have permission of the other, submitting someone else's work as your own is plagiarism, a serious instance of academic misconduct. Everyone is responsible for learning the material themselves. Some of the assignments (programming or homework) may be graded orally.

The numeric grade will be converted into a letter grade according to the following scale

Minimum Score	92	90	87	82	80	77	72	70	67	62	60	0
Grade	A	A-	B+	B	B-	C+	C	C-	D+	D	D-	F

There is no curve, but the instructor reserves the right to increase a grade if he believes it would not reflect the student's mastery of the material. A grade can only be decreased for cheating.

Graduate students will be given alternate homework assignments.

## 6 Special Features of the Filesystem

All your work should be done in the AFS volume assigned to you. The absolute pathname is

```
/afs/cs.uwm.edu/users/classes/cs654/401/PantherID.
```

We recommend that you make a symbolic link from your UNIX home directory to this directory. Before you can access this directory, you will need to obtain authenticate yourself to AFS using Kerberos. This can be done using

```
/usr/afsws/bin/klog PantherID
```

and type your Kerberos password.

The root directory of your volume has some special subdirectories:

**OldFiles** The directory includes a copy of your volume taken at 1am. If you accidentally delete a file that you need, please look for a backup here before contacting lab staff.

**Grades** This directory includes a file giving the grade for each assignment. The first line gives the numeric grade and the remainder of the file gives comments. The file `summary.txt` lists all the scores and computes the weighted percentage.

**homework $n$**  Directory for homework deliverables.

**PA $n$**  Directory for programming assignments.

**misc** Any files you don't need in an assignment.

The file system is a distributed network filesystem that permits you to do your homework on you home computer once you have installed Open AFS. The program assignments require that you have Java and Scala (both freely available) installed. You may do assignments on `weise` or on your home machine, but you will need an environment that enables shell scripts and make files (e.g. Linux).

## 7 Examinations

On Thursdays, there is a quiz on reading except for the three "midterm" examinations:

**Midterm #1** February 23

**Midterm #2** April 5

**Midterm #3** May 10

The final examination is on May 15.

## 8 Schedule

The following schedule is subject to change.

<b>Tuesday</b>	<b>Thursday</b>	<b>Tuesday</b>	<b>Thursday</b>
January 24 <i>What is a compiler?</i>	January 26 <i>The Cool compiler</i> Ch 1 <b>Quiz on Reading</b>	March 27 <i>Assembly Language</i> Ch 5.0–5.4 on CD <b>HW 4 due</b>	March 29 <i>Back-end &amp; Assembling</i> Ch 14.0–5 <b>Quiz on Reading</b>
January 31 <i>Regular Expr. &amp; Automata</i> Ch 2.1.1, 2.2.1 <b>PA 1 due</b>	February 2 <i>Scanners</i> Ch 2.2 <b>Quiz on Reading</b>	April 3 <i>Linking</i> Ch 14.6–7 (inc. CD) <b>PA 5 due</b>	April 5 <b>MIDTERM 2</b> Ch 3,7,9,14
February 7 <i>Grammars and Trees</i> Ch 2.1.2–3 <b>HW 1 due</b>	February 9 <i>Parsing I</i> Ch 2.3.0–2 <b>Quiz on Reading</b>	April 10 <i>Expression Evaluation</i> Ch 6.0–3 <b>HW 5 due</b>	April 12 <i>Selection &amp; Iteration</i> Ch 6.4–5 <b>Quiz on Reading</b>
February 14 <i>Parsing II</i> Ch 2.3.3 <b>PA 2 due</b>	February 16 <i>Error Handling</i> Ch 2.3.4 on CD <b>Quiz on Reading</b>	April 17 <i>Calling Sequences</i> Ch 8.0–8.2 <b>PA 6 due</b>	April 19 <i>Other Issues</i> Ch 8.3–5 <b>Quiz on Reading</b>
February 21 <i>Attribute Grammars</i> Ch 4 <b>HW 2 due</b>	February 23 <b>MIDTERM 1</b> Ch 1–2,4	April 24 <i>Optimization</i> Ch 16.0–2 on CD <b>HW 6 due</b>	April 26 <i>Pipelining</i> Ch 5.5 on CD <b>Quiz on Reading</b>
February 28 <i>Binding &amp; Scopes</i> Ch 3.1–3 <b>PA 3 due</b>	March 1 <i>Symbol Tables</i> Ch 3.4–8 (inc. CD) <b>Quiz on Reading</b>	May 1 <i>Local Optimization</i> Ch 16.3 on CD <b>PA 7 due</b>	May 3 <i>Global Optimization</i> Ch 16.4 on CD <b>Quiz on Reading</b>
March 6 <i>Types &amp; Type Checking</i> Ch 7.0–7.2 <b>HW 3 due</b>	March 8 <i>Type Constructors</i> Ch 7.3,7.4,7.10 <b>Quiz on Reading</b>	May 8 <i>TBA</i> CH 10?,11?,12? <b>HW 7 due</b>	May 10 <b>MIDTERM 3</b> Ch 5,6,8,16
March 13 <i>Object Orientation</i> Ch 9.0–3 <b>PA 4 due</b>	March 15 <i>Object Layout</i> Ch 9.4–5 (inc. CD) <b>Quiz on Reading</b>	May 15 <b>FINAL EXAM</b> Ch 1–9,14–16 <b>PA 8 due</b>	
March 20 SPRING	March 22 BREAK		

## 9 Accomodations

If you will be needing any accomodation in this course for any reason, please contact the instructor. Please also be aware of the standard University policies: <http://www.uwm.edu/Dept/SecU/SyllabusLinks.pdf>