

CISC / CMPE 458 Programming Language Processors
Winter 2020
Preliminary Schedule

Professor J.R. Cordy, Rm. 625 Goodwin Hall, phone 533-6054, email cordy@cs.queensu.ca
Office hours: Wed 14:00-15:00, other times by appointment.

Lectures Wednesday evening 18:30 - 21:30, WLH 210

Tutors Karim Jahed, Joey Sun, Carson Cook

Tutorials Thursday evening 6:30-7:30 pm, Botterell Hall B139, beginning Thursday Jan 16

Advising Monday & Tuesday time TBA, Goodwin Hall 230, beginning Monday Jan 27


References PT: A Pascal Subset, J.A. Rosselet, University of Toronto, 1980.
(These are all reprinted as appendices in the textbook.)

Marking 4 Quizzes 20% Team Project 40% Final Exam 40%

There is no "examination only" option in this course. However, you must pass the final examination to pass the course, and you must get at least B on the exam to get an A in the course.

Quizzes There will be four 20 minute quizzes on the lecture material. Quizzes will be given in class at the beginning of the lecture time. Each quiz will cover the lecture and text material up to and including the immediately preceding class, or as specified. Tentative quiz dates are:
Wed Jan 29 Wed Feb 12 Wed Mar 11 Wed Apr 1

Course Project The course project will consist of implementing a small set of language extensions to an existing compiler for a modest programming language. The project will be done in teams of 4 to be chosen (by you) and registered with the instructor by Wednesday Jan 22.

Projects will be marked teamwise, which is to say each member of the team will receive the team mark for the project part of the course (although part of the mark will be a self-evaluation of the members of your team). It is the team members’ responsibility to insure that work on the project is divided more or less equally among the members of the team. If your team begins to develop problems in this regard, see me immediately. The end of the course is much too late to entertain complaints that members of your team did not do their share of the work. Be sure that every team member has some participation in every phase – examination questions can depend on you understanding every phase of the project.

The project is designed to give you realistic experience in the practice of software engineering as well as providing hands-on experience in building a programming language compiler. The compiler system you will be working with uses the same practical compiler construction technique (S/SL) that is used in some of IBM’s high-quality compilers for PL/I and other languages and is based on the same top down parsing method used in Gnu C, Ruby, Python and many other modern language processors. Thus it serves as a good example of the application of modern techniques of compiler construction.

The compiler has been carefully structured into four functionally independent components, and thus serves as a good example of medium-scale software engineering as well. As part of the project it will be necessary to read, understand and modify each module of the compiler, providing you with real experience in software maintenance, the task that accounts for over 85% of the time of practicing software engineers. Finally, because you will be undertaking the project in teams, you will be faced with a realistic problem in software project management.

The compiler will be implemented in four phases. Project phases will be handed out and due at the beginning of the Wednesday lecture, with tentative start and due dates for each phase on:

<table>
<thead>
<tr>
<th>Phase</th>
<th>Handed out</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Scanner/Screener</td>
<td>Wed Jan 22</td>
<td>Wed Feb 5</td>
</tr>
<tr>
<td>2. Parser</td>
<td>Wed Feb 5</td>
<td>Wed Feb 26</td>
</tr>
<tr>
<td>3. Semantic Analysis</td>
<td>Wed Feb 26</td>
<td>Wed Mar 18</td>
</tr>
<tr>
<td>4. Code Generator</td>
<td>Wed Mar 18</td>
<td>Wed Apr 8</td>
</tr>
</tbody>
</table>

Lateness Policy
Work is normally due electronically, by 4pm on the due date. Work handed in late will not be marked unless accompanied by a convincing explanation.

Computing System & Language
CASlab undergraduate server Linux (linux[123456].caslab.queensu.ca), using PT Pascal, a subset of standard Pascal (yuck!), and the actual compiler you will be working on. Your first problem (starting now) is learning to program in PT Pascal.

Course Outline
I. Introduction, Overview, Basic Concepts, Compiler Structure
II. S/SL - Syntax/Semantic Language
III. Lexical and Syntactic Specification of Programming Languages
IV. Scanning and Screening
V. Parsing
VI. Runtime Model
VII. Semantic Analysis
VIII. Implementing the Runtime Model
IX. Storage Allocation
X. Code Generation
XI. Bytecode Interpretation