## Module Code
CS3071

## Module Title
Compiler Design I

## Pre-requisites
A basic understanding of machine architectures along with a thorough knowledge of programming in both assembly language and in high level programming languages such as C, C#, C++ or JAVA.

## ECTS
5

## Chief Examiner
Dr David Abrahamson

## Teaching Staff
Dr David Abrahamson

## Delivery
<table>
<thead>
<tr>
<th>Lecture hours</th>
<th>Lab hours (per student)</th>
<th>Tutorial hours (per student)</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>11</td>
<td>3</td>
<td>44</td>
</tr>
</tbody>
</table>

**Comments:** Attendance at all lectures, labs and tutorials is compulsory.

## Aims
To teach the principles of compiler design.

## Learning Outcomes
On successful completion of this module students should be able to:
- design lexical analysers based on finite-state automata;
- design top-down parsers based on push-down machines;
- design I-attributed translation grammars to specify the semantic processing of high-level programming language constructs;
- design a recursive-descent parser for any given LL(1) grammar;
- use Coco/R—a compiler generator tool—to construct lexical analysers and recursive descent parsers for high-level programming languages.
## Syllabus

After a brief introduction to the subject, the first few classes will be devoted to the study of finite-state automata and their use in the design of efficient lexical analysers based on reduced (minimal) machines. This will be followed by an in-depth study of syntax analysis (with an emphasis on top-down parsing methods) based primarily on principles of attributed translation. The final few classes will be devoted to the study of some basic code generation techniques along with a selection of associated run-time implementation issues.

Tutorial exercises will be incorporated into lecture periods throughout the course, and students will be introduced to the more practical side of compiler construction during the weekly laboratory sessions.

There will be three significant pieces of graded coursework, each of which will involve aspects of compiler design and implementation.

## Assessment

Examination 80%, coursework 20%.

Assessment the supplemental examination is by written paper which contributes 100% to the overall mark; there is no coursework component.

Examination duration two hours.

## Bibliography

Compiler Design Theory, Philip M Lewis, Daniel J Rosenkrantz and Richard E Stearns, Addison Wesley, 1976


## Website

None