Module Code | CSU22014
---|---
Module Name | Systems Programming
ECTS Weighting | 5 ECTS
Semester taught | Semester 1
Module Coordinator/s | Prof. David Gregg

**Module Learning Outcomes**

On successful completion of this module, students will be able to:

- LO1. Design and construct clear, maintainable, correctly structured programs in C to solve a variety of practical problems
- LO2. Develop programs using UNIX development tools such as vi, gedit, gcc, clang and gdb
- LO3. Distinguish between good, questionable and bad programming style
- LO4. Assess alternative program design decisions
- LO5. Effectively use pointers and dynamic memory allocation
- LO6. Develop code that performs bit manipulation and bit packing/unpacking

**Module Content**

Students taking this module have already successfully completed courses in object-oriented Java programming and ARM assembly language programming. This module starts with a new perspective on program construction, with structured programming in C. This part of the module deals with the features of C, sound design principles for structured programming, and the necessary self-discipline required to program in a low-level language like C. Students also learn the basics of how programs are commonly implemented, with special emphasis on the layout of program data in memory. In parallel to learning C programming, the students learn the basics of the Unix Operating System.

**Teaching and Learning Methods**

Students attend two lectures and one lab each week. There is lab work almost every week, and an optional tutorial for students who need additional assistance.

**Assessment Details**

<table>
<thead>
<tr>
<th>Assessment Component</th>
<th>Brief Description</th>
<th>Learning Outcomes Addressed</th>
<th>% of total</th>
<th>Week set</th>
<th>Week due</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examination</td>
<td>Two hour written examination</td>
<td>LO1, LO2, LO3, LO4, LO5, LO6</td>
<td>80%</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>Lab work</td>
<td>Weekly lab work that builds to three pieces of submitted work</td>
<td>LO1, LO2, LO3, LO4, LO5, LO6</td>
<td>20%</td>
<td>Weekly</td>
<td>Three due dates</td>
</tr>
</tbody>
</table>

**Reassessment Details**

Examination (2 hours, 100%)

---

1 TEP Glossary
2 TEP Guidelines on Workload and Assessment
### Contact Hours and Indicative Student Workload

<table>
<thead>
<tr>
<th>Contact Hours (scheduled hours per student over full module), broken down by:</th>
<th>44 hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>lecture</td>
<td>22 hours</td>
</tr>
<tr>
<td>laboratory</td>
<td>11 hours</td>
</tr>
<tr>
<td>tutorial</td>
<td>11 hours</td>
</tr>
<tr>
<td>other</td>
<td>0 hours</td>
</tr>
</tbody>
</table>

**Independent study (outside scheduled contact hours), broken down by:**

| preparation for classes and review of material (including preparation for examination, if applicable) | 36 hours |
| completion of assessments (including examination, if applicable) | 36 hours |

**Total Hours**: 116 hours

### Recommended Reading List

- The C Programming Language, Brian Kernighan and Dennis Ritchie
- The Practice of Programming, Brian Kernighan and Rob Pike
- Programming Pearls, Jon Bentley

### Module Pre-requisites

**Prerequisite modules:** CSU11010, CSU1012, CSU11021, CSU11022

**Other/alternative non-module prerequisites:** Two semesters of introduction to programming, and some experience of assembly programming

### Module Co-requisites

CSU22011

### Module Website


### Last Update

19/09/2019 by David Gregg