# COMP1911 Course Introduction

Note that the content of this course is different from previous offerings of COMP1911. This course gives an introduction to programming and is suited to any student who is interested in computing but is still deciding whether they will undertake further study in the area.

---

**Aims**

The objective of this course is for students to develop proficiency in programming in the C programming language and to develop a background of relevant knowledge and skills on which to base further study of computing.

Topics covered include:

- fundamental programming concepts
- the C programming language
- the C compiler
- programming style
- program design and organisation concepts
- program testing concepts
- program debugging concepts
- the C debugger
- program documentation
- systematic problem solving

Practical experience of these topics is supplied by laboratory programming exercises and assignments.

**Learning Outcomes**

After completing this course, students should be able to design, construct, document and test reasonably complex C programs that accurately comply with a given specification. Further, more advanced, C programming topics will be covered in the follow-up course, COMP1921. In addition, students should have also acquired some familiarity with the Unix/Linux operating system and tools, the C compiler, and the C debugger.

**COMP1911 vs COMP1917**

This course is intended for students who are **not** computer science majors. If you are a computer science major you should **not** be enrolled in this course unless you have a good reason, which you have discussed with your degree coordinator.

Computer science majors and others who have a keen interest in computer science should enroll in COMP1917 instead!

**Units of Credit (UoC)**

6 units

**Prerequisites**

There are no pre-requisites for this course.

**Course Web Site**

http://www.cse.unsw.edu.au/~cs1911

**Handbook Entry**

http://www.handbook.unsw.edu.au/undergraduate/courses/2008/COMP1911.html

**Staff**

| Staff Name | Role | E-mail |
|---|---|---|
| Dr. Salil Kanhere | Lecturer-In-Charge | salilk "at" cse.unsw.edu.au |
| Ms. Angela Finlayson | Subject Administrator | cs1911 "at" cse.unsw.edu.au |

**Reading Materials**

The recommended text for this course is:

Alistair Moffat, *Programming, Problem Solving, and Abstraction with C*, Pearson Educational, Australia, 2003, ISBN 1-74103-080-3.

The following books might also serve as additional reference material:

Brian W. Kernighan and Dennis M. Ritchie, *The C Programming Language*, 2nd edition, Prentice Hall, USA, 1988, ISBN 0-13-110370-9.

Paul Davies, *The Indispensable Guide to C*, Addison Wesley, England, 1995, ISBN 0-201-62438-9.

**Lectures**

These are the lecture times and locations:

| Time | Location |
|---|---|
| Mon 11am-12noon | CLB 6 |
| Wed 9am-10am | CLB 6 |
| Wed 2pm-3pm | CLB 6 |

The following table is a tentative schedule of the lecture topics and assessment tasks. Students will be informed of any changes during the lectures and by announcements on the notices page.

| Week | Lecture Topics | Assessment Tasks | Lab and Tutes |
|---|---|---|---|
| 1 | Course Introduction, Introduction to C, Types, Variables | Assignment 1 released | Lab 1 & Tute 1 |
| 2 | Operators, I/O, If statement | | Lab 2 & Tute 2 |
| 3 | Debugging, Program Development | | Lab 3 & Tute 3 |
| 4 | Char type, while loops | Assignment 2 released | Lab 4 & Tute 4 |
| 5 | Arrays, for loops, strings | Assignment 1 due | Lab 5 & Tute 5 |
| 6 | Functions | Project released | Lab 6 & Tute 6 |
| 7 | Design, Testing, Debugging | | Lab 7 & Tute 7 |
| 8 | Pointers | | Lab 8 & Tute 8 |
| 9 | Pointers, Command line arguments, header files | | Lab 9 & Tute 9 |
| 10 | Design, Separate Compilation, Make | Assignment 2 due | Lab 10 & Tute 10 |
| 11 | Static variables, Type casts, generic pointers | | Lab 11 & Tute 11 |
| 12 | Revision | Project due | Lab 12 & Tute 12 |

**Tutorials**

Tutorials give you a chance to clarify ideas mentioned in lectures and to practice your problem-solving skills in a small (and hopefully more personal) class with the assistance of a tutor. You should make sure that you use them effectively by examining in advance the material to be covered in each week's tutorial, by asking questions, by offering suggestions and by generally participating. The tutorial questions will be posted on the Web in the week before each tute. Tutorials are one hour long and they are scheduled to start in Week 1 of session. There will often be a couple of extra questions, which are not covered during the tutorial. You may use these for practice or revision.

**Laboratory Classes**

Laboratory Classes give you a chance to *practice* programming skills on small, simple examples. The examples have been chosen to highlight particular aspects of programming, and will give you enough grounding in programming practice to enable you to complete your project work relatively easily. Your tutor will be present in your Lab Class to answer any questions you may have.

All lab class exercises must be shown to your tutor during your scheduled lab class. The lab exercises will be available during the session via the course's Web page. *Students are advised not to fall behind in their laboratory work*. Each lab exercise is worth 2 marks. There will be some opportunities to get some bonus marks. Note that, the total lab marks will be capped at 20 marks. In order to get a mark for a lab exercise you must show your tutor

a completed or partially completed solution before the end of your scheduled lab class in the week that the exercise is scheduled. If you cannot complete the exercise by the end of the lab you may complete it in your own time and submit it using the `give` command **before** your next lab class. Your tutor will then mark it at the very beginning of the next scheduled lab class.

**Note:** To gain a mark you must at the very least: show your tutor a partial solution; and, submit a completed solution using `give` before your next lab class. *After that time, you will not receive a mark for that particular lab exercise*. If you are unable to attend your scheduled lab class due to illness or misadventure, then you should apply for special consideration and an extension may be granted. You can only have your lab work marked during your scheduled lab class. It is not possible to have your work marked in another class.

To make sure that you can complete the work in your scheduled lab class, you should make an effort to prepare the work beforehand. You can make use of the Lab simply by booking a time-slot. You have sufficient bookable Lab time available each week to allow you to prepare your lab exercises; make good use of it.

Your weekly lab class is 2 hours long and it starts in Week 1 of session. Where possible, your lab class will be scheduled to follow on from your weekly tutorial and will have the same tutor (but you will need to walk to a new room).

**Assignments**

There will be three programming assignments. Assignments give you the chance to practice what you have learnt on relatively large problems (compared to the small exercises in the labs). Assignments are a very important part of this course, therefore it is essential that you attempt them yourself.

| Task | Weighting |
|---|---|
| Deliverable 1 | 5% |
| Deliverable 2 | 10% |
| Group Project | 15% |

The percentages represent percentages of your final mark for this course.

**Late Submission**

It is often difficult for students to catch up once they have fallen behind. So you are strongly encouraged to complete all of your programming assignments on time.

Late programming assignments will be penalised unless you have legitimate reasons to convince the LIC otherwise. Any assignment submitted after the due date will attract a reduction of 10% per day to the maximum mark. A day is defined as a 24-hour day and includes weekends and holidays. Precisely, an assignment that is submitted X hours after its due date is considered to be ⌈X/24⌉ days late. *No assignment will be accepted more than five days after its due date.*

In the event of an illness or a family emergency that makes it impossible for you to complete an assignment on time, extensions may be granted beyond its due date. To request an extension for an assignment, contact the LIC *no later than 24 hours before the assignment is due*.

**Plagiarism**

All work submitted for assessment *must be your own work*. Lab exercises and assignments *must be completed individually*. We regard copying of assignments or lab exercises, in whole or part, as a very serious offence. *We use plagiarism detection software to search for suspiciously similar submissions. This software is a lot better at detecting similarities than you might expect!*

Collaborative work in the form of "think tanking" is encouraged, but students are not allowed to derive code together as a group during such discussions. Students are also warned not to send code fragments of the assignments or labs to each other in any form (e.g., as e-mail or listings). In addition, copying/purchasing of code that is available on the Web is also not permitted. Students who are singled out during our regular plagiarism sweep will be dealt with according to School Policy, which provides serious penalties particularly in the case of repeat offenses:

> http://www.cse.unsw.edu.au/~studentoffice/policies/yellowform.html
> http://www.cse.unsw.edu.au/help/doc/primer/node42.html
> http://www.cse.unsw.edu.au/~chak/plagiarism

Students are strongly advised to protect their work. Do not leave your terminal/computer unattended, or leave listings at the printer for others to take.

**Overall Assessment**

The assessable components of the course are:

| Component | Mark | Type |
|---|---|---|

| | | |
|---|---|---|
| Lab Exercises | 10% | practical component |
| Assignments | 30% | practical component |
| Written Exam | 60% | written component |

All submitted programs (lab exercises and assignments) must adhere to the Course Style Guide.

IMPORTANT

The written and practical components of this course are related but qualitatively different and an all-round good performance is required to pass this course.

Your final mark will be the **weighted harmonic mean** of your exam mark and the total of your practical marks, except that if your exam mark is less than 50% your final mark for the subject will be your exam mark.

Your final mark will be calculated as follows:

$$\textbf{Final mark} = (PM \times EM) / (0.6 \times PM + 0.4 \times EM)$$

where, PM = lab marks + assignment marks and EM = exam mark, both expressed as percentage.

Where your practical marks are significantly higher than your exam mark, your practical marks may be reduced and you may be required to attend an interview to explain the difference.

Exam marks and final marks may be scaled to ensure that the subject Pass/Fail boundary and the Distinction/High Distinction boundary reflect a consistent standard from session to session.

**Supplementary Exams**

The document "Important Advice for Students" states the supplementary assessment policy for the School of CSE. Please take the time to read it carefully. If you are granted a Supplementary Examination, then it is most likely to be held in December. If you think that you may be eligible for a supplementary exam, then make sure you are available at that time. It is your responsibility to check at the School Office for details of Supplementary Examinations.

**Getting Help**

Since for many of you this is your first time studying computing at university, you are likely to have quite a few questions about a variety of things.

If you do have a question please follow the chain of enquiry below to ensure a timely response. Please note that as you move up the chain the number of students that person looks after increases and the likelihood of receiving an immediate response decreases.

1. check the subject or CSE forums for an answer to your question
2. post a question on the subject or CSE forums
3. ask one of your COMP1911 peers
4. ask your tutor
5. attend a consultation session

For administrative issues please contact the subject administrator.

The School Web site has useful information that may be of help:

> http://www.cse.unsw.edu.au/

Particularly useful is the Unix `man` pages. The searchable interface allows you to find out more about various Unix commands (this assumes that you have some basic knowledge of the Unix environment):

> http://www.cse.unsw.edu.au/scripts/man

If you are new to the computing laboratory and its setup, you can read about it in the *Unix Primer*. The *Unix Primer* and other useful documentation can be found at this link:

> http://www.cse.unsw.edu.au/help/doc/index.html

Take the time to explore the system. The more adventurous you are, the more you will learn about the system and the more proficient you will get. Those interested in working from home should look into purchasing a copy of our *Home Computing CD*, which is available from the CSG Help Desk. It contains a collection of programs and utilities used by various courses within the School. To find out more, go to:

> http://www.cse.unsw.edu.au/~homecomputing

One final piece of advice, if you have a problem, do not wait until Week 12 before you seek help. Problems that are dealt with early are usually quicker and easier to resolve than if they are left to the last minute.

**Course Feedback and Improvement**

The previous CATEI evaluation indicated that students were generally happy with the way the course was run. As a result, we are maintaining the same structure during the current session. One suggestion from the students was to include some additional challenging extensions to programming assignments. In this offering, we will add such extensions to one of the assignments.

---