COMP2100

Systems Programming

Session 2, In person-scheduled-weekday, North Ryde 2026

School of Computing

Contents

General Information

Unit convenor and teaching staff
Unit convenor
Kate Stefanov
kate.stefanov@mq.edu.au

Lecturer
Richard Han
richard.han@mq.edu.au

Lecturer
Lachlan Patrick
lachlan.patrick@mq.edu.au

Credit points
10

Prerequisites
COMP1010

Corequisites

Co-badged status
COMP6100

Unit description
This unit studies the boundary between software and the systems on which software executes. It considers the impact of constraints imposed by operating systems and hardware systems on the design and performance of computer software. Students will learn how to operate within these constraints to build effective systems software. The unit studies these issues by looking below the abstractions provided by high-level programming languages. The unit is an introduction to systems programming and related issues. It provides an entry point into more advanced study of computer systems in the following areas: hardware implementation, operating systems, network design and programming, and programming language design and implementation.

Learning in this unit enhances student understanding of global challenges identified by the United Nations Sustainable Development Goals (UNSDGs) Industry, Innovation and Infrastructure

Important Academic Dates

Information about important academic dates including deadlines for withdrawing from units are

available at https://www.mq.edu.au/study/calendar-of-dates

Learning Outcomes

On successful completion of this unit, you will be able to:

ULO1: Use a command line interface to control the computer.

ULO2: Produce code that utilises system software to provide controlled access to system resources.

ULO3: Analyze machine representations of data, such as integer and floating point.

ULO4: Analyze assembly code fragments. Trace execution and re-express with higher level descriptions.

ULO5: Identify security flaws that can arise from program execution.

ULO6: Explain how the hardware impacts on software performance and how software can interact directly with the hardware.

General Assessment Information

Grading Requirements

The assessment has three components: the two programming tasks, and the final exam. The final mark for the unit will be calculated by combining the marks for all assessment tasks according to the percentage weightings shown in the assessment summary.

Requirements to Pass this Unit

To pass this unit you must:

- Achieve a total mark equal to or greater than 50%

Programming Tasks (Labs)

There are two individual programming tasks to be completed during the semester. These tasks are to be your own work; **the use of Generative AI is not allowed.** These tasks develop your skills and assess your progress. These programming tasks are to be done in your own time, but you may also use workshop time to work on them and to seek assistance from the teaching staff. They are due as specified below.

The programming tasks are designed to be completed over several weeks and are split in stages. Do not leave them until the last week! The automated submission software keeps track of the submission dates for each stage. As part of the total mark for a programming task, cumulative learning progress marks are awarded for each stage submitted on time before the expected completion date for that stage as secified both on iLearn.

Release Dates

The Data File Lab will be released no later than 31 July (the Friday of Week 1).

The Binary Bomb Lab will be released no later than 11 September (the Friday of Week 7).

Both Programming tasks are Individual Work

Both The Data File Lab and The The Binary Bomb Lab are set as individual work and each one is personalised for a particular student.

The University's academic integrity policy will be enforced. You may assist your fellow students with general concepts, pointers to resources and useful tools or commands that are publicly available. You may **not** become involved in any way in helping a fellow student to find the solution to their particular task, nor may you share with them any aspect of the solution of your particular task. If you decide to develop or modify a tool (including software tools, procedures or methods) to assist you in solving your programming task, you may not provide that tool to your fellow students, nor may you publish it. Each practical task will include additional specific instructions to help you understand what is acceptable and what is not.

Each programming task must be the sole work of the student turning it in. Any cheating will be handled under the University's Academic Integrity Policy.

The following are guidelines on what collaboration is allowed for programming tasks and what is not [adapted from CS:APP website]

What is Cheating?

- *Sharing code or other electronic files*: either by copying, retyping, looking at, or supplying a copy of a file from this or a previous semester. Be sure to store your work in protected directories, and log off when you leave any lab computer, to prevent others from copying your work without your explicit assistance. Don't use a public repository (such as github) for your programming task files.
- *Sharing solutions, quizzes or exams*: Looking at, copying, or supplying a programming task solution, quiz or exam.
- *Sharing procedures, methods or tools* that have been developed to solve challenges of a programming task. If you wish, you may develop your own procedures, methods or tools to assist you in the tasks, but you may not share them with others, publish them or store them in a public repository such as github. If you find someone else's procedures, methods or tools, you may not use them -- if in doubt, ask the teaching staff of your registered class or the unit convenor. For example, you may not provide or use somebody else's sequence of steps that can be followed to solve (or partially solve) a programming task. Similarly, you may not provide or use somebody else's program that is designed to solve (or partially solve) a stage of a programming task.
- *Using other's code:* Using code that you did not write yourself. You may not use code from courses at other institutions, or from any other non-COMP2100 source (e.g., software found on the Internet, such as ChatGPT). You may, however, use the Internet and other sources to learn useful concepts, ideas, and programming idioms. If you find helpful ideas on the Internet or in other sources, you should acknowledge them in

comments in your code. For Internet sources, a URL is sufficient acknowledgement.

- *Looking at other's code.* Although mentioned above, it bears repeating. Looking at other students' code or allowing others to look at yours is cheating. There is no notion of looking "too much," since no looking is allowed at all.

What is NOT Cheating?

- Clarifying ambiguities or vague points in class handouts or textbooks.
- Helping others use the computer systems, networks, compilers, debuggers, profilers, or other system facilities.
- Helping others with high-level design issues.
- Helping others with high-level (not code-based) debugging techniques.
- Using code from the CS:APP website or from the COMP2100 iLearn pages.
- Learning programming idioms and techniques from examples.
- Reading Unix manual pages, forums, etc in order to find out how to perform particular tasks (e.g. set a breakpoint in a debugger) or use programming language/library features (such as printf).
- Asking for help from the practical demonstrator, teaching staff or lecturer. You may also seek help from the School of Computing's Drop-In Centre.

Be sure to store your work in protected directories, and log off when you leave any lab computer, to prevent others from copying your work without your explicit assistance.

Programming Task Submissions

Submission instructions for programming tasks will be provided on iLearn. The two tasks contain differing forms of automatic assessment and feedback tools that will assist you during the task. The automated feedback will help you identify your mistakes, but does not replace your own analysis of the problem, debugging and testing. The automated assessment contributes strongly to our understanding of your performance in the task.

Each programming task will specify a particular due date. The due time will be 11:55 pm on the due date unless otherwise specified.

For the programming tasks you are given free three days extensions in total. You can apply these to any due date including stages.

Important Note: These extensions are distinct from the standard university short-extension policy.

- You must apply for these days via the lab command in the assignment software.
- Please be aware that standard short-extension requests are not permitted in COMP2100.

- The total allowance for both programming tasks combined is three days.

You do not have to apply for special consideration to get these. See iLearn and the command line manual for each programming tasks for instructions on how to do that.

In-person assessments

The coding skills and knowledge you gain from the programming tasks will be evaluated through in-person assessments during your Week 7 (Data File Lab) and Week 13 (Binary Bomb Lab) workshops.

You are required to attend the in-person assessment during your registered workshop session. If you are unable to attend your usual workshop, and you have been granted a special consideration, you must attend the in-person assessment session organised on Saturday, 19 September for The Data File Lab or Saturday, 14 November for The Binary Bomb Lab, respectively. **There will be no other dates available to sit your in-person assessments.**

Submissions

Submissions in The Data File Lab are made through the lab command; submissions in The Binary Bomb Lab are made automatically as you progress through the task of analysing and executing the "binary bomb" program.

Late submissions and Special Consideration

If you experience serious and unavoidable difficulties that affect your ability to meet the due dates for progress or the closing date of a programming task, you may apply for special consideration as explained at https://students.mq.edu.au/study/my-study-program/special-consideration. If the request is accepted, the action may be to grant an extension of the relevant due date(s), or it may be to require you to submit an alternative assessment item. Extensions, if granted, are managed through the automated assessment system that you access via the lab command.

If you apply for special consideration, please note:

- Apply promptly. Late applications may make it impossible to sensibly offer an extension, and you may risk having to complete a different assessment task which would mean starting from scratch. For example, if you are ill for two days just before the due date, an extension of two days would be reasonable, but that extension cannot be granted more than two days after the due date since the extension end date would have already passed!
- Contact your TA, the senior TA or lecturers if you have questions about extensions.

Late Assessment Policy

For any late submission of time-sensitive tasks, such as scheduled tests/exams, performance assessments/presentations, and/or scheduled practical assessments/labs, students need to submit an application for Special Consideration.

Our Late policy is summarized below.

Assessments where Late Submissions will be accepted

In this unit, late submissions will be accepted as follows:

- Data File Lab - NO, unless Special Consideration is Granted (after the free three extension days as previously described in the section on Programming Task Submissions)
- Binary Bomb Lab - NO, unless Special Consideration is Granted (after the free three extension days as previously described in the section on Programming Task Submissions)
- Exams - NO, unless Special Consideration is Granted

Assessment Tasks

Name	Weighting	Hurdle	Due	Groupwork/Individual	Short Extension	AI Approach
Data File Lab	30%	No	05/09/2026	Individual	No	Observed
Binary Bomb Lab	30%	No	31/10/2026	Individual	No	Observed
Final Examination	40%	No	Exam period	Individual	No	Observed

Data File Lab

Assessment Type [1]: Experiential task
Indicative Time on Task [2]: 30 hours
Due: **05/09/2026**
Weighting: **30%**
Groupwork/Individual: Individual
Short extension [3]: No
AI Approach: Observed

Develop C programs to manipulate data files.

On successful completion you will be able to:

- Use a command line interface to control the computer.
- Produce code that utilises system software to provide controlled access to system resources.
- Analyze machine representations of data, such as integer and floating point.

Binary Bomb Lab

Assessment Type [1]: Experiential task

Indicative Time on Task [2]: 30 hours
Due: **31/10/2026**
Weighting: **30%**
Groupwork/Individual: Individual
Short extension [3]: No
AI Approach: Observed

Use debugging tools to understand the behaviour of a precompiled program. Find input strings that will be acceptable to the program (i.e. defuse the binary bomb).

On successful completion you will be able to:

- Use a command line interface to control the computer.
- Analyze machine representations of data, such as integer and floating point.
- Analyze assembly code fragments. Trace execution and re-express with higher level descriptions.
- Identify security flaws that can arise from program execution.

Final Examination

Assessment Type [1]: Examination
Indicative Time on Task [2]: 26 hours
Due: **Exam period**
Weighting: **40%**
Groupwork/Individual: Individual
Short extension [3]: No
AI Approach: Observed

Written examination

On successful completion you will be able to:

- Use a command line interface to control the computer.
- Produce code that utilises system software to provide controlled access to system resources.
- Analyze machine representations of data, such as integer and floating point.
- Analyze assembly code fragments. Trace execution and re-express with higher level descriptions.
- Identify security flaws that can arise from program execution.
- Explain how the hardware impacts on software performance and how software can interact directly with the hardware.

---

[1] If you need help with your assignment, please contact:

- the academic teaching staff in your unit for guidance in understanding or completing this type of assessment
- Academic Success for academic skills support.

[2] Indicative time-on-task is an estimate of the time required for completion of the assessment task and is subject to individual variation.

[3] An automatic short extension is available for some assessments. Apply through the Service Connect Portal.

Delivery and Resources

Week One

In Week 1 we introduce the unit, go over the unit guide, and begin delivering technical content in the lecture and workshops. Workshops start in Week 1.

Text Book

"Computer Systems: A Programmer's Perspective" 3rd edition, R.E. Bryant and D.R. O'Hallaron, Pearson 2015.

You are required to read set sections of the text book each week. The text book is available in electronic form in the library, or you can purchase a printed copy from a book seller of your choice.

Recommended Text

"The C Programming Language" 2nd edition, Brian W Kernighan and Dennis M Ritchie, Prentice-Hall 1988.

This small book is the classic reference on C programming.

iLearn Web Site

All learning materials will be published on iLearn including lecture slides, programming tasks and other explanatory documentation. We will also be making key announcements via iLearn as well as answer questions in the unit's Forum.

Lectures

Lectures are a core learning experience where we will discuss the theoretical underpinnings and concepts that are essential to this unit. Key ideas for the programming tasks will be discussed from time to time in lectures.

Lectures will be provided in person. Participation in lectures is not required but is highly recommended. Lecture recordings will be provided on echo360.

Workshops

Each week it is important to attend your workshop. We will be using the workshop to cover

important concepts that may be useful to solving weekly questions, programming/lab issues, and/or exam questions. Workshops also provide an opportunity to discuss specific questions related to any aspect of the unit: the lecture content, the programming tasks and the set practical questions.

Workshops provide an opportunity for you to develop your skills in systems programming and your understanding of the key concepts of the unit. Workshops combine discussion with practical programming experience. Your workshop teaching staff will provide you with individual assistance and may also from time to time address the entire class to provide useful information to assist with programming tasks.

We also hold in-reson assessments for the two programming tasks in workshops. You are expected to sit these in the workshop you are enrolled in. Please note that failing that may result in a reduced mark for the respective programming task assessment. For details see the section on Late submissions and Special considerations.

Unit Forum

A forum for unit discussions is provided on iLearn. Students are free to post questions, comments or hints in relation to any aspect of the unit, except that you should avoid posting any questions, hints, comments or solutions that could be interpreted as cheating (see above). Civility is essential in all such postings and interactions.

Don't post your code, or anyone else's code, on the forums!

If you see a post from another student that appears to be cheating, please email the senior TA (to be specified on iLearn), and we will remove the offending post.

General Communication and Contact

Apart from contacting teaching staff in-person before/after the lecture and/or workshops, and at a pre-arranged time, the main way of communication is via the iLearn forums. For general administrative questions relating to the unit, personal queries, etc. please email the convenor, the lecturers, and the Senior TA for the unit at COMP2100@mq.edu.au

For specific technical questions, first contact your TA, use the Unit Forum and/or use either the announced weekly consultations time for the Senior TA in COMP2100, or the School of Computing's Drop-In Centre. If your technical question still cannot be resolved, email the convenor.

Unit Schedule

The detailed unit schedule will be available on iLearn. The unit is roughly organised as follows:

Week 1-6: C programming, Command line interface, Data representations (integer and float), some system API.

Weeks 7-12: Assembly code and how C programs appear in the machine, Operating System features and implementation.

Policies and Procedures

Macquarie University policies and procedures are accessible from Policy Central (https://policies.mq.edu.au). Students should be aware of the following policies in particular with regard to Learning and Teaching:

- Academic Appeals Policy
- Academic Integrity Policy
- Academic Progression Policy
- Assessment Policy
- Fitness to Practice Procedure
- Assessment Procedure
- Complaints Resolution Procedure for Students and Members of the Public
- Special Consideration Policy

Students seeking more policy resources can visit Student Policies (https://students.mq.edu.au/support/study/policies). It is your one-stop-shop for the key policies you need to know about throughout your undergraduate student journey.

To find other policies relating to Teaching and Learning, visit Policy Central (https://policies.mq.edu.au) and use the search tool.

Student Code of Conduct

Macquarie University students have a responsibility to be familiar with the Student Code of Conduct: https://students.mq.edu.au/admin/other-resources/student-conduct

Results

Results published on platform other than eStudent, (eg. iLearn, Coursera etc.) or released directly by your Unit Convenor, are not confirmed as they are subject to final approval by the University. Once approved, final results will be sent to your student email address and will be made available in eStudent. For more information visit connect.mq.edu.au or if you are a Global MBA student contact globalmba.support@mq.edu.au

Academic Integrity

At Macquarie, we believe academic integrity – honesty, respect, trust, responsibility, fairness and courage – is at the core of learning, teaching and research. We recognise that meeting the expectations required to complete your assessments can be challenging. So, we offer you a range of resources and services to help you reach your potential, including free online writing and maths support, academic skills development and wellbeing consultations.

Student Support

Macquarie University provides a range of support services for students. For details, visit http://students.mq.edu.au/support/

Academic Success

Academic Success provides resources to develop your English language proficiency, academic writing, and communication skills.

- Ask a Study Coach
- Access StudyWISE
- Upload an assignment to Studiosity
- Complete the Academic Integrity Module

The Library provides online and face to face support to help you find and use relevant information resources.

- Subject and Research Guides
- Ask a Librarian

Student Services and Support

Macquarie University offers a range of Student Support Services including:

- IT Support
- Accessibility and disability support with study
- Mental health support
- Safety support to respond to bullying, harassment, sexual harassment and sexual assault
- Social support including information about finances, tenancy and legal issues
- Student Advocacy provides independent advice on MQ policies, procedures, and processes

Student Enquiries

Got a question? Ask us via the Service Connect Portal, or contact Service Connect.

IT Help

For help with University computer systems and technology, visit https://students.mq.edu.au/support/technology/service-desk.

When using the University's IT, you must adhere to the Acceptable Use of IT Resources Policy. The policy applies to all who connect to the MQ network including students.

Changes from Previous Offering

We value student feedback to be able to continually improve the way we offer our units. As such we encourage students to provide constructive feedback via student surveys, to the teaching staff directly, or via the FSE Student Experience & Feedback link in the iLearn page.

Student feedback from the previous offering of this unit was very positive overall, with students pleased with the clarity around assessment requirements and the level of support from teaching staff. As such, no change to the delivery of the unit is planned, however we will continue to strive to improve the level of support and the level of student engagement.

Changes since First Published

Date	Description
17/07/2026	Clarified the distinction between the unit-specific three-day extension provision and the standard university short-extension policy

Unit information based on version 2026.02 of the Handbook