UNSW College

Course Outline

January 2024

STEM Diploma

DPST1091 / CPTG1391

Introduction to Programming

1. Staff

Position	Name	Email
Course Convenor & Lecturer	Pantea Aria	P.Aria@unswcollege.edu.au

2. Course information

Units of credit (UOC): 6

Pre-requisite(s): none

Total course contact hours: 96

2.1 Course summary

This course introduces students to the basics of programming. Topics covered include:

- fundamental programming concepts
- the C programming language and use of a C compiler
- programming style
- program design and organisation concepts
- program testing and debugging

The course does not assume any previous programming experience.

2.2 Course aims

From recent innovations in AI like self-driving cars to humanoid robotics navigating complex environments, leapfrogs in battery technology to sequencing the human genome - the world is benefiting and evolving thanks to computer systems. At the core of all these systems are computers executing instructions to solve exciting problems.

In this course, you will learn the fundamentals of how we instruct computers to solve problems. You will explore the architecture and mechanics of how computers operate and how you can translate real-world problems to computer programs that solve these problems.

The concepts you learn in COMP1511/DPST1091/CPTG1391 will provide a foundation for your future endeavours in computing and, we hope, will begin to change the way you think about real-world problems.

The course aims for students to become proficient in programming using a high level language, C. By the end of the course, students should be able to construct C programs to solve problems.

2.3 Course learning outcomes (CLO)

At the successful completion of this course you (the student) should be able to:

1. understand the core syntax & semantics of the C programming language including types, I/O, arrays, functions, pointers, structs, file manipulation and dynamic memory allocation
2. given a problem, solve it by proficiently constructing (designing, testing, debugging) a secure, reliable, and correct C program
3. understand & employ fundamental data structures including linked lists
4. use Linux and Unix-like operating systems to develop and test software

2.4 Relationship between course and program learning outcomes and assessments

Course Learning Outcome (CLO)	**Program Learning Outcome (PLO)**	**Related Tasks & Assessment**
CLO 1	Conceptual understanding of computer underpinnings (EA1.2)	Exam, Labs, Assignments, Tutorial Performance
CLO 2	Understanding of specialist bodies of engineering knowledge (EA1.3)	Exam, Labs, Assignments, Tutorial Performance
CLO 3	Understanding of underpinnings (EA1.1)	Exam, Labs, Assignments, Tutorial Performance
CLO 4	Understanding of specialist bodies of engineering knowledge (EA1.3)	Labs, Assignments

3. Strategies and approaches to learning

3.1 Learning and teaching activities

This course involves several teaching activities:

Lectures – 4 hours per week

Lectures present theory and concepts, by way of case studies and practical examples. Lecture notes will be provided in advance of each class. There will be 4 hours of timetabled lectures each week. Lecture notes will be available on the course web pages before each lecture.

Tutorials – 1 hour per week

- Attendance will be taken for all Tutorial sessions.
- Tutorials allow students to collaboratively work through example problems to illustrate lecture idea and have concepts from lectures clarified by the tutor. ***In every tutorial from Week 2 onwards, students will be given a short question to work through and submit during their tutorial. Students must attempt and submit at least 70% of the assigned tutorial assessments to pass the course. This is a hurdle for the course.***

Lab Classes – 3 hours per week

- Attendance will be taken for all Lab sessions.

Lab A Sessions:

- These lab classes involve small exercises where students build systems that illustrate the ideas covered in lectures. In **some** of the 2 hour Lab A classes, students lab Problem Sets may be done in pairs or small groups, and you and other students can work through them together, learning from each other. Lab demonstrators will facilitate you forming pairs/groups during Laboratory Sessions.
- There are also some Challenge Exercises for anyone looking to push beyond the course content or see some interesting continuation of the content in the course. They are not necessary to complete to obtain full marks.
- If you cannot complete any exercises by the end of the lab you may complete them in your own time and submit them using the "give" command before 9am Monday (Monday 9:00 AEST) in the week after the lab. For each day after that time, the maximum mark it can achieve will be reduced by 5% (off the ceiling). Labs will be marked automatically around a week after the due date.
- Students will seek help from their lab demonstrators during lab A/B sessions.
- You cannot obtain marks by e-mailing lab work to tutors.

- Problem Sets (lab exercises) have an indicator to help you choose which problems to do -- you should prioritise the one-dot exercises first, then two-dot, then three. You must complete all one-dot and two-dot exercises to get full marks in the problem set for the Week. Any three-dot exercises you complete will form extra bonus marks to compensate for any other missing marks in the problem set component.
- There will be a **Practice Exam** towards the end of term in lecture time.
- The total marks for the Practice Exam and Problem sets are capped at 15 ***marks*** (there are 4 possible bonus marks from the three-dot exercises that could bring you up to a total of 15 if you missed out on any other marks in the one- or two-dot exercises). ***Completing just the one- and two-dot exercises every week*** can give you the full 15 marks needed in this component.

Indicator	Description	Marks total per week
●○○	These exercises examine the most fundamental or crucial skills in the course. Every student in the course should complete these exercises to ensure they finish the term with the skills necessary to complete the exam and future courses. *If you are having trouble completing these exercises, please contact us or come along to the help session, so that we can help you.*	0.7
●●○	Every student in the course should attempt these. These questions may take slightly longer to solve than the questions of one dot importance; and may apply fundamental skills in more involved ways but doing them will help to link together concepts taught in lectures and tutorials with the practical components. *Being able to solve these questions will be helpful in the exam, assignments, or in future courses.*	0.7
●●●	These exercises are for students who want to be exposed to more complex applications of the content they've been taught. These exercises usually require a lot more time to solve and complete. *Being able to solve these questions is helpful in the later stages of the assignment (Stages 3 and 4 of each assignment) and later questions of the final exam (Questions 6 onwards).*	0.4
❀	These exercises are for students who want to extend themselves. They may not necessarily teach you more programming but will expose you to difficult problems and interesting parts of Computer Science, where you will get a good opportunity to practice some killer problem-solving skills!	no marks, only for challenge.

Lab B Sessions:

In the 1 hour Lab B classes, students will work on revision exercises or can get help completing lab A or assignment work. Lab B revision exercises are not worth any marks.

Because this course is practical in nature, laboratory classes are a very important component. If you do not put a great deal of effort into the lab classes, you risk failing the final exam. Please use laboratory sessions to ask tutors for any help you may need if you are stuck on that week's problems.

Assignments – 2 during the course

There are two assessable programming assignments. Assignments give you the chance to practice what you have learned 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. Collaboration with other students is limited to discussion of fundamentals, not any discussion of assignment specifics. All assignments are carefully checked for plagiarism, which can result in serious academic consequences.

- Assignment 1 (Submission in Week 8 Friday 9am) 20%
- Assignment 2 (Submission in Week 12 Friday 9am) 25%

Late assignments submissions will be penalised. The exact penalty will be specified in the assignment specification - typically it is a reduction in mark by 5% per day late for 5 days, at which point the maximum mark drops to 0.

Technical issues and assignment extensions - extensions will only be given on assignment due dates if there is a scheduled server maintenance planned during the working time of the assignment, or in extreme circumstances when servers unexpectedly go down during business hours (8am-10pm). Please make sure to start your assignments early and work on them consistently throughout the term.

Generative AI Permission Level For Assignments

Simple Editing Assistance

In completing this assessment, you are permitted to use standard editing and referencing functions in the software you use to complete your assessment. These functions are described below. You must not use any functions that generate or paraphrase passages of text or other media, whether based on your own work or not.
If your Convenor has concerns that your submission contains passages of AI-generated text or media, you may be asked to account for your work.

DPST1091/CPTG1391/COMP1511 Specific Information

You are permitted to use the tools **dcc-help** and **dcc-sidekick** to help you understand the error messages you may get when compiling the code you have written.

You are permitted to use **autotest-help** to help you understand why your code may not be passing the automated tests.

You are not permitted to submit code generated by automatic AI tools such as Github Copilot, ChatGPT, Google Bard in DPST1091/CPTG1391/COMP1511 for assignments. Submitting code generated by Github Copilot, ChatGPT, Google Bard and similar tools will be treated as plagiarism.

Our reasoning behind our decisions:

Systems such as Github Copilot and ChatGPT based on large language models or other generative artificial intelligence techniques, look likely to become heavily used by programmers. However, you need a good understanding of the language you are coding in and the systems involved before you can effectively use these tools. Using these tools to generate code for **DPST1091/CPTG1391/COMP1511** instead of writing the code yourself will hinder your learning.

Online Forum

We will be using teams for our online forum, which allows students to ask and answer questions on the tutorial, lab, and assignment exercises, and on lecture material.

Final Exam

The format of the final exam is yet to be finalised. The format and mode of the exam will be advised closer to the end of the term. It will contain implementation tasks that will require you to write C programs. It will also contain sections which require you to read code or answer questions to show your knowledge of programming.

During this exam you will be able to execute, debug and test your answers. The implementation tasks will be like those encountered in your weekly labs.

We will provide you with sample questions in the last week of the course.

Generative AI Permission Level in the Final Exam

No Assistance

This assessment is designed for you to complete without the use of any generative AI. You are not permitted to use any generative AI tools, software or service to search for or generate information or answers.

COMP1511 Specific Information

You will not have access to any generative AI tools during the exam.

You will not have access to dcc-help, dcc-sidekick or autotest-help during the exam.

3.2 Consultations

There will be consultation sessions starting in week 2 where tutors will be available for one on one help with specific problems and assignment clarification. These sessions are optional and will run at a scheduled time during the week, with more sessions available around assignment deadlines and during exam study period. Check the course timetable for what consultations have been scheduled.

Consultations will be held online or face-to-face through Blackboard Collaborate.

3.3 Expectations of students

Students are expected to:

- attend all lectures, and ask questions, but otherwise not disturb other students.
- attend all tutorials and actively participate in the discussions.
- attend all lab classes and work diligently on the exercises.
- do all the assignment work themselves, asking only the forum or tutors for help.

On the course forum, students should:

- use relevant/meaningful message titles on all posts.
- ask questions clearly and provide sufficient background information that the question can be reasonably answered.
- not post significant pieces of code, especially code for assignments. Course forum staff can view recent autotests and submissions of your work.

4. Course schedule and structure

This course consists of 8 hours of class contact hours per week. You are expected to take an additional 5 hours outside classes to complete assessments, readings, and exam preparation.

Week	**Lectures**	**Tutorial and Labs**	**Assessment**	Related CLO
Week 1	Introduction to course/Linux/C; data types; variables, simple I/O, expressions, If Statements	Setting up working from home (VLAB), Basic Input/Output, Create/run first C programs on Linux.		1,4

Week 2	Structs, Enums, Loops and code Style	Variables and If Statements		1,2,4
Week 3	Functions and Static Arrays	Structs, Enums and Looping		1,2,4
Week 4	Pointers, Dynamic Arrays and Memory	Functions, Arrays		1,2,4
Week 5	Debugging, Strings, Character functions and multi-file projects	Functions, Arrays and Pointers	**Assignment1** Released	1,2,4
Week 6	Dynamic memory allocation and extra C (for loops, pre/post increment, multi-file compilation)	Working with characters and strings		1,2,4
Week 7	*Flexibility Week*	*Flexibility Week*		1,2,4
Week 8	Command line arguments, Memory allocation and Linked Lists	Characters, Strings and Command line arguments	**Assignment1 Due** Friday 9am	1,2,3,4
Week 9	Linked Lists	Programming with Linked Lists	**Assignment2 released**	1, 2, 3, 4
Week 10	Recursion including recursion with linked lists	Programming with Linked Lists, recursion, and Practice Exam		1, 2, 3, 4
Week 11	Revision	Revision		1, 2, 3, 4
Week 12	Exam Information and Revision	Revision	**Assignment 2 Due** Friday 9am	1, 2, 3, 4

5. Assessment

5.1 Assessment tasks

Assessment task	Length	Weight	Due	CLOs
Assessment 1: Programming assignment (Looping and arrays)	3 weeks	20%	Friday week 8	1, 2, 4
Assessment 2: Programming assignment (linked data structures)	4 weeks	25%	Friday week 12	1, 2, 3, 4
Assessment 3: Lab exercises and Practice exam	Throughout the Term	15%	Weeks 2 to 12	1, 2, 3, 4
Tutorial Performance **Hurdle**	During Tutorial	0%	Weeks 2 to 12	1, 2, 3
Assessment 4: Final Exam		40%	Exam period	1, 2, 3, 4

Hurdles

Exam Hurdles:

- Hurdle requirement **#1:** in the final exam you must solve a task by writing a program that uses an **array**.
- Hurdle Requirement **#2:** in the final exam you must solve a task by writing a program that uses a **linked list**.

Tutorial Performance Hurdle:

"***In every tutorial from Week 2 onwards, students will be given a short question to work through and submit during their tutorial. Students must attempt and submit at least 70% of the assigned tutorial assessments to pass the course."***

You **cannot** pass COMP1511/DPST1091/CPTG1391 unless you achieve **ALL** the above hurdles.

Final Mark

Your final mark for this course will be computed using the above assessments as follows:

CourseWorkMark	=	LabMark + Ass1Mark + Ass2Mark	out of 60
ExamMark	=	ExamPracMark + ExamTheoryMark	out of 40
HurdlesOK	=	Pass (**Exam** List Hurdle & **Exam** Array Hurdle & Tutorial Performance Hurdle)	true/false
FinalMark	=	CourseWorkMark + ExamMark	out of 100
FinalGrade	=	UF, if ! HurdlesOK && FinalMark ≥ 50 FL, if FinalMark < 50/100 PS, if 50/100 ≤ FinalMark < 65/100 CR, if 65/100 ≤ FinalMark < 75/100 DN, if 75/100 ≤ FinalMark < 85/100 HD, if FinalMark ≥ 85/100	

5.2 Assessment criteria and standards

In all programming work, the primary assessment criterion is correctness (i.e. does the code produce the expected output/behaviour according to the exercise specification). This will be tested by executing code against a variety of test cases, some of which are available to students, and others of which are used after submission for assessment purposes. Code is also expected to be expressed clearly, with consistent formatting and using relevant variable names. Code style will be assessed in the two assignments of this course.

5.3 Submission of assessment tasks

All assignments will be submitted online via CSE's submission system. Late assignments submissions will be penalised. The exact penalty will be specified in the assignment specification - typically it is a reduction in mark by 5% per day late for 5 days, at which point the maximum mark drops to 0.

If you are unable to submit an assignment by the due date, due to medical reasons or other reasons which significantly affect your ability to carry out your work, you should contact the lecturer as soon as possible, preferably well before the assignment deadline. If the lecturer considers that your ability to complete the assignment on time has been adversely affected, an extension may be granted to make up for the time you were unable to work on the assignment.

Lab exercises must be submitted by the specified due date of the exercises, likely on the Monday.

5.4. Feedback on assessment

Assignments will be marked after the submission deadline and annotated with comments by the tutor. You can discuss the tutor's comments in a lab class after you have received the feedback.

You may seek feedback from your lab demonstrators for your lab submission during the lab class in the week following the submission.

6. Readings and resources

There is no requirement for a textbook for COMP1511/DPST1091/CPTG1391. Generally, students do not purchase this textbook. It covers material in a different way to the COMP1511/DPST1091/CPTG1391 course materials and goes into differing levels of content than this course. It may, however, be useful as a reference, or to explore some content in more detail.

The optional textbook for the course is: Programming, Problem Solving, and Abstraction with C by Alistair Moffat , ISBN 978 1 74103 080 3, which can be purchased from the UNSW Bookshop.

7. Course Evaluation and Development

At the end of every term, COMP1511/DPST1091/CPTG1391 students are invited to provide their feedback about the course through the UNSW myExperience online survey system. This is used to assess the quality of the course to make on-going improvements. We do take this feedback seriously and use it to improve the course materials and their delivery. Students are also encouraged to provide informal feedback during the session, and to let the lecturer in charge or any of the course staff, know of any problems as soon as they arise. Suggestions will be listened to very openly, positively, constructively, and thankfully, and every reasonable effort will be made to address them. Recent MyExperience evaluations showed that students were highly satisfied with most aspects of the course. However, there are always things that can be improved, some changes that we are making this term:

- We have adjusted and restructured some content in the course. We are eliminating some topics that we don't feel are completely necessary to the core aims of COMP1511/DPST1091/CPTG1391 and introduced others in some circumstances (like enumerations).

- This term, we have broken down the assessments, to have an extra assignment in the first three weeks. This also means that we are removing the weekly tests component. This helps us to streamline to only one problem set due each week.

- Last term, we introduced assignment stages - these were positively received as they broke down the assignments into more approachable chunks, whilst still allowing us to achieve our learning goals with the overarching assignment. This term, the stages are sticking around - but all stages will be accessible for each assignment - meaning if you are looking to work ahead, you can! A win for self-paced learning.