

DPST1091 / CPTG1391

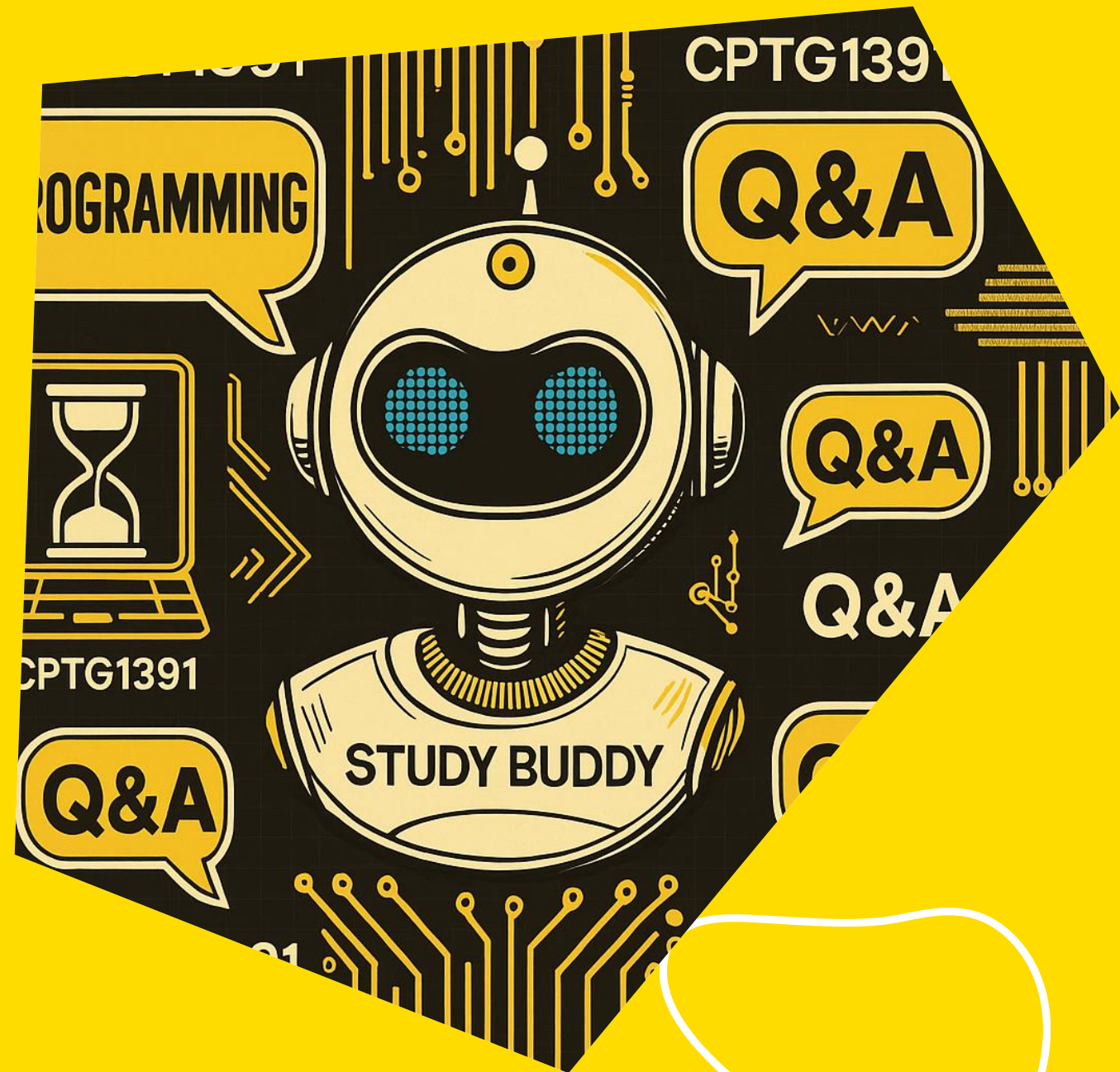
Introduction to Programming

Week 11 Lecture 2

Lecturer and Course Convener:

Dr Pantea Aria

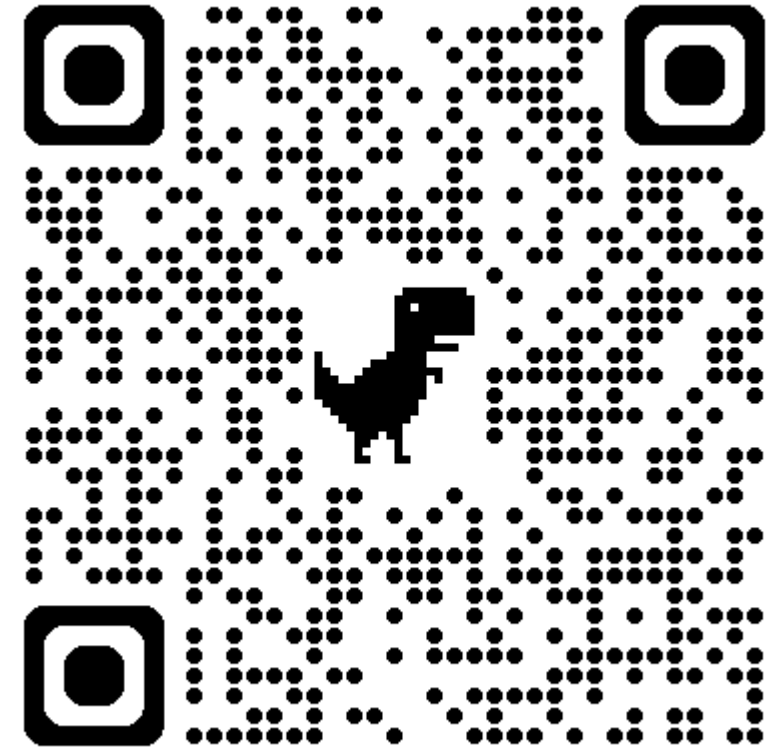
Exam Information



Agenda

- **Today**
 - **Exam Information**
 - **Sample hurdle questions**

Link to the Live Lecture Code



<https://cgi.cse.unsw.edu.au/~dp1091/26T2/code/11>

Live lecture code is written for teaching, not perfection.
It may include extra comments and may not always follow
ideal coding style.

MyExperience Survey

Tell us what **worked** (and what **didn't!**) — I genuinely value your feedback





Exam Information



When and Where

Date

Monday 10th of August 2026

Time

9:00 AM, Please be there at least 30 minutes early

Location

CSE Labs, Your exact building and room will be announced on teams before the exam.

https://taggi.cse.unsw.edu.au/FAQ/CSE_lab_map/

Duration

3 hours + 10 minutes reading time

What is the Final Exam

**Variables: int, double, char,
structs**

**Simple IF statements and WHILE
loops**

Arrays, 2D Arrays

Strings

Pointers

Linked Lists

Final Exam Environment

Same as Practice Exam

dcc, autotests, submit (like give) are the same as in your labs

Submit as many times as you want - only last submission will be marked

There will not be

dcc-help
dcc-sidekick
autotest-help

Exam is closed book, however:

The course website will be accessible during the exam, so that you can access:

Lecture materials and code

Tutorial materials and code

Laboratory materials and code

During the Exam

Exam conditions apply!!!

If you experience any issues during the exam, please raise your hand and wait for an invigilator.

No discussion of the exam or sharing your code with anyone except for DPST1091/CPTG1391 staff

Do not communicate with anyone about the exam within 24 hours of the exam start time - this is considered plagiarism

What can or can't I bring to the exam

You may bring a **clear water bottle**

Pens are allowed

No electronic devices or phones

No paper is permitted – a sheet will be provided for rough work

This sheet will be collected but not marked

You may request additional paper if needed

In the room

Once you enter the room and take your seat, instructions will be provided on how to begin the exam.

All exam code will already be available in your home directory when you log in.

This setup will be the same as the practice exam

You do not need to fetch the exam

You may re-fetch individual question files if needed

Fit to Sit Policy:

By sitting the exam on the scheduled assessment date, you are declaring that you are fit to do so and cannot later apply for Special Consideration.

What if I feel unwell during the exam?

Stop working on the exam and immediately raise your hand.

Let the invigilator know you are unwell.

Submit your work and be escorted from the exam room to discuss with the course convenor/lecturer/admin

Obtain a medical certificate within 24 hours of the exam date

Submit a Special Consideration application within 3 working days saying that you felt ill during the exam and were unable to continue.

Exam Hurdles

There are **array hurdle questions**

(Q2 and Q4)

You must score **at least 50% in one** of these

There are **linked list hurdle questions**

(Q1 and Q3)

You must score **at least 50% in one**

Hurdle questions will be **clearly labelled** in the exam

Exam Structure

The exam contains **11 programming questions**

Questions are rated with **difficulty levels (●, ●●, ●●●)**

Expected progression:

- Most students → complete Q1–Q8
- Many students → attempt Q9

Marks Breakdown

Q1–Q4 (12 marks each) → Hurdles

Q5–Q8 (5 marks each) → Debugging
Focus on understanding core concepts
and fixing code

Q9–Q10 (11 marks each)

Q11 (10 marks)

General Question Style

Similar to **revision exercises and labs**

Difficulty shown using dot rating system

Some questions include **starter code/frameworks**

You must:

- write
- compile
- test your code

Autotests are available

- Harder questions → fewer autotests

Questions 1 & 2 (●○○) Easier

Similar to early
practice questions

Q1 Linked List:

Work with linked lists (int/char/double)

No insertion/deletion required

Q2 Array:

Work with arrays (int/double/struct)

Skills tested:

Variables, structs

scanf, printf

if statements, while loops

Example Q1

Example - Linked List

Find the **largest value** in a linked list

Return **-1 if empty**

```
int largest(struct node *head);
```

Example – Two Linked Lists

Compare two lists

Return **difference in number of nodes**

```
int difference(struct node *head1,  
struct node *head2);
```

Example Q2

Example – Array of Structs

Loop through array and collect information

Sum values for a specific direction

```
int direction_count(int size, struct  
direction array[]);
```

Example – Array Counting

Count how many values are **multiples of 3**

```
int array_count_multiples(int size,  
int array[]);
```

Questions 3 & 4 (●●○) Medium

More complex versions of
Q1 & Q2

May involve:

- multiple loops
- inserting/removing nodes
- 2D arrays
- tracking multiple conditions

Questions 9 & 10 (●●●) Hard

Advanced array manipulation

May include:

- string handling (fgets)
- dynamic memory (malloc, free)
- command line arguments

Question 11
(●●●)
Very
Challenging

Designed for **high-achieving students (HD+)**

Covers **any course content**

Tests ability to **break down complex problems**

Marking

Q1–Q4, Q9–Q11 → manually marked

Q5–Q8 → auto-marked only

Important:

- Passing autotests ≠ guaranteed marks
- Extra hidden tests are used

Code Style

No marks for style

However:

- code must be readable
 - helps debugging and marking
-

What to Study

Focus on **fundamentals**:

- variables, structs, enums
- loops, functions
- arrays, linked lists

Also understand:

- strings
- pointers
- memory allocation

Practice:

- lab exercises
- revision exercises
- practice exam

Advice:

- don't rely on AI tools during study
-

Login Details

Memorise your **zID** and **zPass**

No access to devices or notes
during exam

Exam Tips

Read all questions first

Start with easier ones

Carefully follow instructions

Plan before coding

Draw diagrams (especially for linked lists)

Strategy Advice

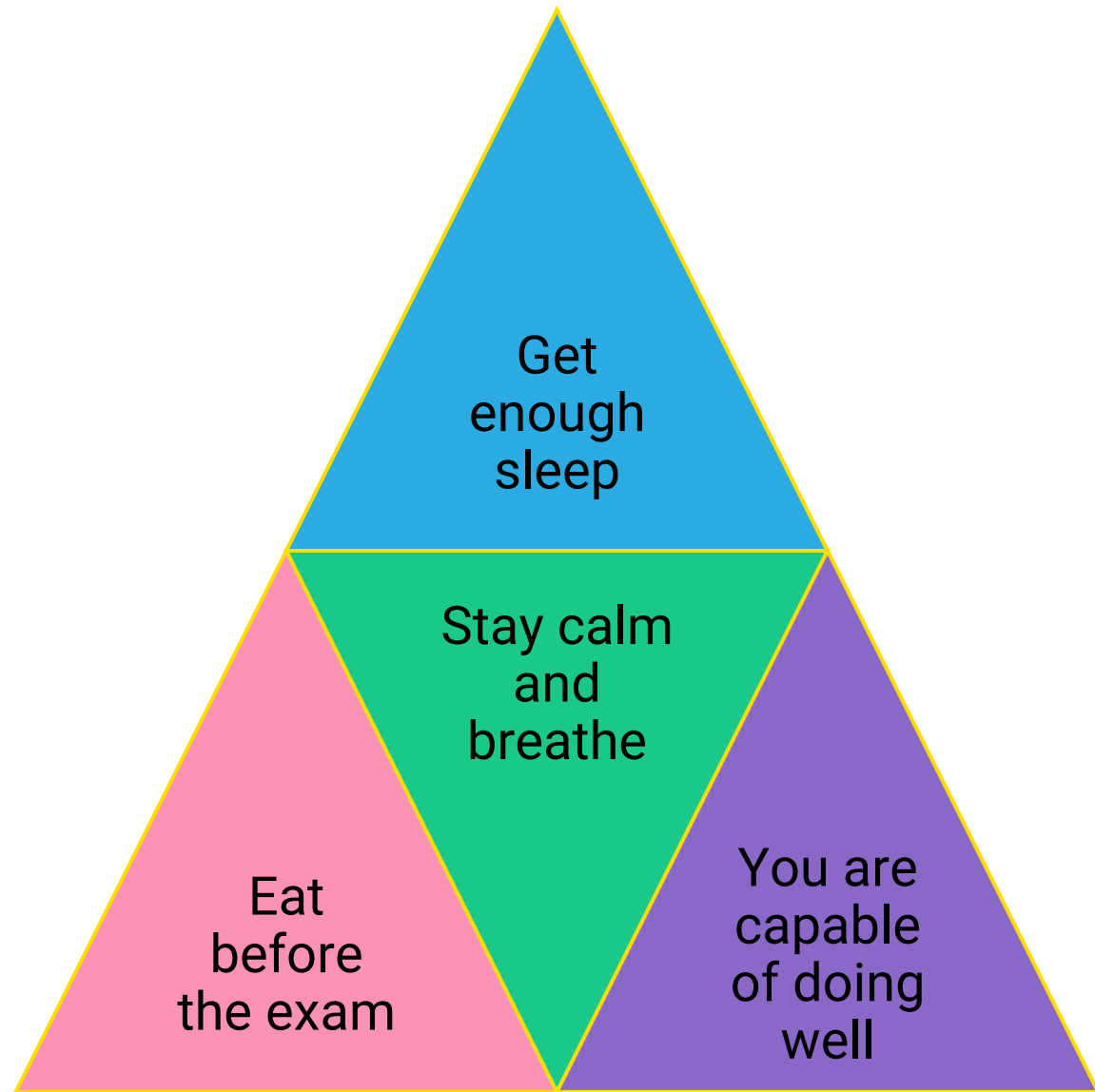
Fully complete fewer questions
partially doing many

Don't get stuck too long on one
question

Later questions are very difficult

You don't need to complete all to
get a high mark

General Advice



Revision

Sample Hurdle Exam Questions

Linked List Hurdle Example (●○○)

File: linked_list_largest.c

Given a linked list, determine the **largest value** stored in the list.

If the list is empty, return -1 .

```
int largest(struct node *head);
```

Linked List Hurdle Example (●○○)

File: linked_list_difference.c

Given two linked lists, calculate the **difference in the number of nodes** between them.

```
int difference(struct node *head1, struct node *head2);
```

Linked List Hurdle Example (●○○)

File: linked_list_equal.c

Given two linked lists, count how many values in the **first list** are equal to the values in the **same position** in the second list.

```
int equal(struct node *head1, struct node *head2);
```

Array Hurdle Example (●○○)

File: array_count_multiples.c

Given an array of integers, count how many elements are **multiples of 3**.

```
int array_count_multiples(int size, int array[]);
```

Array Hurdle Example (●○○)

File: array_step_count.c
Given an array of the following struct:

```
struct steps {  
    int number;  
    char direction;  
};
```

Calculate the **total number of steps** taken in a specified direction.

```
int step_count(char dir, int size, struct steps array[]);
```

Voice of the Student

Anonymous ongoing feedback
Anything you wanted to share with me



Voice of the Student



[Voice of the Student – Fill out form](#)

See you soon ...