COMP1511/1911 Programming Fundamentals

Week 10 Lecture 2

The Final Lecture

Last Lecture

- Exam Revision
  - Array Hurdles
  - String Programming Question
  - The Final Kahoot

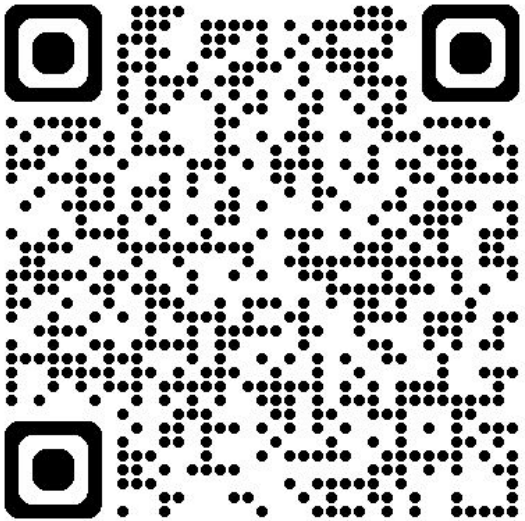
Today’s Lecture

- More exam revision questions
  - Hurdles
  - String, command line argument questions
- Q&A with the tutors

There are many more questions than we will cover in the lecture.
The rest can be used for your own revision.

Link to Week 10 Live Lecture Code

https://cgi.cse.unsw.edu.au/~cs1511/26T3/code/week_10/



Revision and Drop In Sessions

- **Revision Sessions:**
  - Last set of revision sessions on in week 11
  - See forum post for more details, bookings and to vote on topics!
- **Drop in Sessions**
  - Email cs1511@unsw.edu.au to book a drop in session.
- More practice exams being released at the end of this week

My Experience Surveys



http://myexperience.unsw.edu.au/

🪪 Don't forget to bring your physical student ID card!

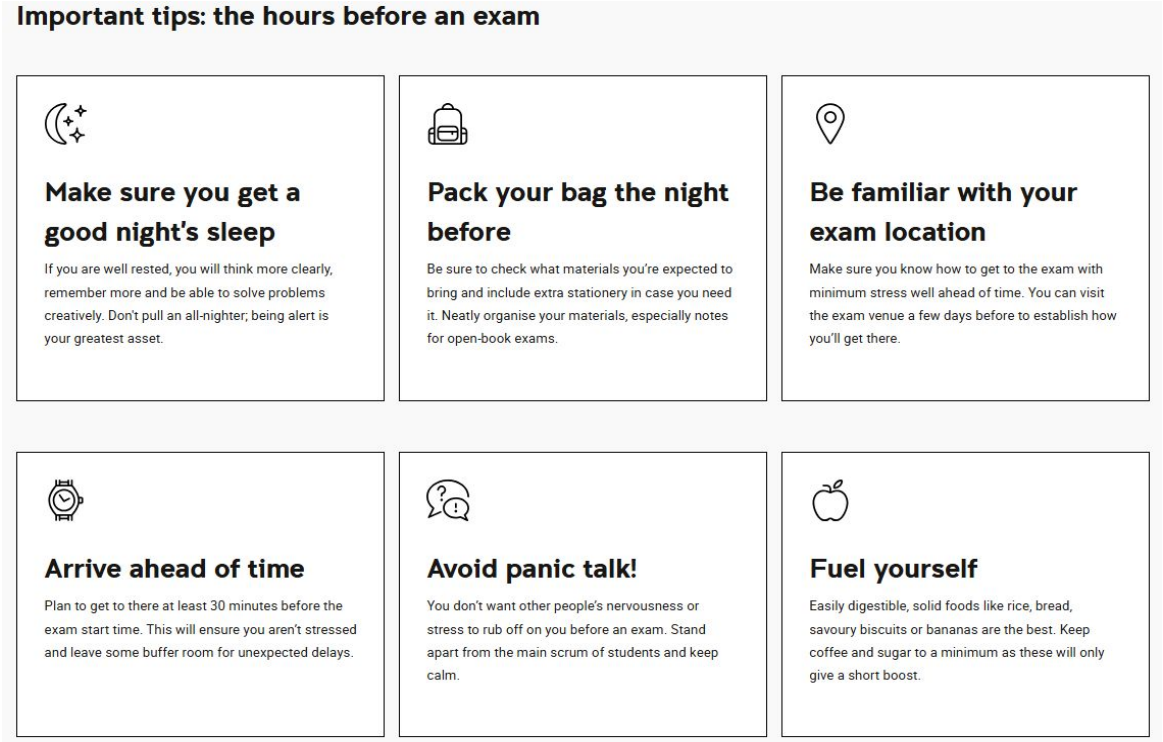
You must display your **UNSW student identification card at the door of your exam venue** to be scanned, and then on your desk for the **duration of the exam**.

If your name has changed or your card has expired or is lost or damaged, book an appointment with The Nucleus: Student Hub immediately for a replacement ID card .

⚠️ You will not be able to sit your exam without physical photo identification.

Students must carry their UNSW student ID whenever they are on campus. You should also carry an official government-issued photo ID as a backup. For it to be accepted as backup exam identification, it must be a physical copy and include an English component. Acceptable documents include a driver's licence, NSW photo card or passport.

Wellness for Exams



More information can be found here: [Exam preparation | Current Students – UNSW Sydney]

Debug Questions

Debug List (●○○)

debug_list.c

The following code should find the minimum value in a list and return the value -1 if the list is empty:

```
$ ./debug_list 4 10 5 2 9
$ 2
$ ./debug_list
$ -1
```

Debug Example (●○○)

debug_string.c

Note: you may NOT use the strcpy or strcat function.

The following code is meant to join two strings together and form one string. For example:

```
$ ./debug_string
$ Enter the first string: Last
$ Enter the second string: Lecture
LastLecture
```

Sample One Dot Hurdle Exam Questions

Linked List Hurdle Example (●○○)

`linked_list_largest_odd.c`

Perform some computation or comparisons on a linked list

Given a linked list, return the largest odd value in that list.
If the list is empty, or contains no odd values return -1
Assume all values in the list >= 0

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

COMPLETED IN WEEK 9 TUESDAY LECTURE

Linked List Hurdle Example (●○○)

`linked_list_difference.c`

Perform some computation or comparisons on 2 linked lists

Given 2 linked lists return the difference in the number of nodes in each list. You should return the absolute value of the difference.

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

Linked List Hurdle Example (●○○)

`linked_list_num_equal.c`

Given two linked lists, return the number of values in the first linked list that are equal to the corresponding values in the second linked list.

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

Linked List Hurdle Example (●○○)

`list_find_divisible.c`

Count all the elements in the linked list that are divisible by a given value and output the count.

```
int find_divisible(int value, struct node *head);
```

// To find elements in list 6->1->7->12->NULL that are divisible by 3 run like:

```
./list_find_divisible 3 6 1 7 12
2
```

Linked List Hurdle Example (●○○)

`linked_list_range.c`

Find the range (the difference between the biggest term and the smallest term) of a linked list

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

Linked List Hurdle Example (●○○)

`list_even_difference.c`

Given two linked lists, count the number of even numbers in both linked lists and return the difference.

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

Array Hurdle Example (●○○)

`array_count_multiples.c`

Given an array:
Count the number of multiples of 3 in the array.

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

Array Hurdle Example (●○○)

array_step_count.c

An array of structs of type:

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

return the total of the number of steps taken in a given direction

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

Array Hurdle Example (●○○)

array_struct_capitals.c

An array of structs of type:

```
struct initials {
    char first_initial;
    char last_initial;
};
```

return the count of all structs that don't have capital letters for both of their initials.

```
int capitals(int size, struct initials array[]);
```

COMPLETED IN WEEK 10 MONDAY LECTURE

Array Hurdle Example (●○○)

array_count_odd.c

Given two arrays, return the number of items at corresponding indexes that are odd in both arrays.

```
int count_odd(int size1, int arr1[],
              int size2, int arr2[]);
```

Array Hurdle Example (●○○)

`array_multiplied.c`

Write a C program that reads integers into an array from terminal until a number is entered which when multiplied by another number previously entered results in 56. E.g.

```
./array_multiplied
2
3
28
28 * 2 = 56
```

Array Hurdle Example (●○○)

`array_odd_even.c`

Write a C program that reads integers from standard input until it reads a negative integer.

It should then print the odd numbers on one line and then print the even numbers on the next line.

You can assume a maximum of 1000 integers are read before a negative integer is read

Array Hurdle Example (●○○)

```
$ ./array_odd_even
1
2
3
2
-42
Odd numbers were: 1 3
Even numbers were: 2 2
```

Array Hurdle Example (●○○)

`array_closest_to_target.c`

In this task, you are given an array of integers with exactly size elements, and a target number. Your function closest_to_target should return the sum of any two numbers in the array that is closest to the target number.

You can assume that size will be >= 2.

`COMPLETED IN WEEK 10 MONDAY LECTURE`

Two Dot Hurdles

Linked List Hurdle Example (●●◌)

`linked_list_duplicate.c`

Duplicate every node in the list by inserting the same node after the original node.

```
struct node *duplicate(struct node *head);
```

COMPLETED IN WEEK 9 TUESDAY LECTURE

Linked List Hurdle Example (●●◌)

`linked_list_delete_duplicate.c`

Delete the first instance of a duplicate in the given list e.g.

1->3->5->3->1->7->X would give
1->3->5->3->7->X

```
struct node *delete_duplicate(struct node *head);
```

Linked List Hurdle Example (●●◌)

`list_insert_run_sum_node.c`

After every two nodes, insert a new node whose value is the sum of the previous 2 nodes;

```
struct node *insert_run_sum_node(
    struct node *head
);
./list_insert_run_sum_node 3 1 2 5
[3, 1, 4, 2, 5, 7]
```

Linked List Hurdle Example (●●◌)

`list_delete_div.c`

delete the first node that is divisible by the given value

```
struct node *delete_div(int value,
                        struct node *head);
```

// To delete first value divisible by 3 in list 2->9->1->6->NULL

```
./list_delete_div 3 2 9 1 6
[2, 1, 6]
```

Array Hurdle Example (●●◌)

`array2d_min_row_sum.c`

Write a C program that finds the sum of the minimum numbers in each row in a 2D array.

```
int min_row_sum(int size,
                int array[NUM_ROW][NUM_COL]);
```

COMPLETED IN WEEK 10 MONDAY LECTURE

Array Hurdle Example (●●◌)

`array_max_subarray_sum.c`

Write a C program to find the largest sum of contiguous subarray in an array.

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

COMPLETED IN WEEK 10 MONDAY LECTURE

Non-hurdle programming questions

Command Line Arguments (●●●)

common_argument.c

Write a program to print out the most common command line argument and print out the number of occurrences e.g

$./common_argument hooray no hooray hooray potato

```
Most common argument is hooray and it occurs 3
times
```

$./common_argument

```
There are no arguments
```

COMPLETED IN WEEK 10 MONDAY LECTURE

Substring (●●●)

substring.c

Write a C program that reads in words from standard input until Ctrl+D, and checks whether a word read in as a command line argument is a substring in the word. If it appears, print it again. For example:

```
$./substring courage age bloom encourage
blooming
blooming
encouraging
wage
wage
```

Isogram (●●●)

`isogram.c`

An isogram is a word, in which no letter of the alphabet occurs more than once. Write a C program that reads in strings until Ctrl+D, and checks whether each string is an isogram.

For example:

```
$./isogram
Hello
This is not an isogram
```

Reflecting on the Term

- If you were starting the course again, what would you do differently?
- What is the most important thing you learnt in the course?
- What are you most proud of achieving in the course?



C Coding

- Cons
  - Is not the easiest language to learn
  - Makes you do a lot of things yourself!
- Pros
  - Gives you a better understanding of how computers actually work
  - Gives you very fine-grained control
  - You get to play around with pointers and malloc and free
  - Provides a strong foundation for learning other programming languages

This course is about more than just learning C!

Problem Solving

Problem Solving

- Understanding the problem
- Breaking the problem down
- Developing a systematic approach to solving problems
- Trying different approaches when things don’t work
- Perseverance

These skills can be applied to non-technical areas too!

Testing and Critically Evaluating Code

- Gaining experience with test suites
  - Importance of testing a range of inputs, including edge cases
  - Identifying errors and investigating their causes
- Critically evaluating and testing solutions
  - Will it pass the unseen auto-marking tests?
  - Is it making any unjustified assumptions?
- This is more important than ever in era of genAI
  - Just because ChatGPT generates a solution doesn't mean it's correct!

These skills are essential across engineering, science and other technical fields!

Technical Communication and Professional Practices

- Reading and interpreting specifications
  - Sounds simple… but as you've discovered, it isn't!
  - In the real world, requirements are incomplete or ambiguous
  - Asking good questions to clarify
- Asking and answering technical questions in tutorials, labs, lectures, help sessions, check-offs and on forums
- Following style guides and agreed conventions
  - Producing clear, understandable work
- Getting code reviewed by tutors
  - Learning from feedback and improving your work

Time Management

- Planning ahead and managing deadlines
- Prioritising tasks and breaking work into manageable steps
- Allowing time for testing and revision and breaks!
- Knowing when to seek help
  - Recognising when you are stuck
- Allowing time to seek help
  - Hard to get help at midnight the night before the due date



Q&A with the tutors

Your chance to hear/ask about some or all of the following

- Exam tips
- What to enrol in next term
- Getting internships and jobs
- Being a tutor



Thanks and Goodluck! 

A big thanks to all the amazing staff in the course…

the Admins, the Lecture Moderators, Assignment writers, Forum staff, Revision Session staff, Help Session staff and Tutors!

And of course a huge congratulations and thank you to all of the amazing students who have worked hard throughout the term and answered and asked so many great questions!!!

We hope you have learnt a lot!

Reach Out

Content Related Questions:

Course Forum

Admin related Questions email:
cs1511@unsw.edu.au



Student Support - I Need Help With...

My Feelings and Mental Health Feeling depressed, overwhelmed or not your usual self?	**Mental Health Connect** student.unsw.edu.au/**mhc** **TalkCampus** student.unsw.edu.au/**Talkcampus**	**UNSW (24/7) Mental Health Support:** +61(2) 9385 5418 **Text support After Hours**: 0485 826 595 **Offshore** **Call 24-hour Medibank Hotline:** +61 2 8905 0307
Uni and Life in Australia Stress, financial, visas, Accommodation & More	**Student Support** **Indigenous Student Support**	student.unsw.edu.au/**advisors** nura-gili-centre-indigenous-programs
Reporting Sexual Assault/Harassment	**Equity Diversity & Inclusion (EDI)**	edi.unsw.edu.au/**sexual-misconduct**
Educational Adjustments To manage my studies and disability / health condition	**Equitable Learning Services (ELS)**	student.unsw.edu.au/**els**
Academic and Study Skills	**Academic Skills**	student.unsw.edu.au/**skills**
Special Consideration Because life impacts our studies and exams	**Special Consideration**	student.unsw.edu.au/**special-consideration**
Physical Health **Doctors visits, Vaccinations**	**Health Service**	student.unsw.edu.au/**hsu**