What's in the Exam?

Meme credit: Hyperbole and a Half

The Exam - Timing and details

1pm (Australian Eastern Standard Time) Wednesday 18th August
13:00-19:00 (6 hours)

- Completed on your own computer at home
- Files will be provided to you when you run certain terminal commands
- Autotest and Submission will also be done with terminal commands
- Expected to take around 3-5 hours
- You can submit more than once, your final submission will be marked
- Your Week 10 labs will show you what the exam environment is like
- Before and during the exam, we will contact you via your UNSW email if/when we need to

The Exam - “Open Book”

“Open Book” - you can use any resources from the class or online

- Still an exam! No communication between students!
- Any communication about the exam within 24 hours of the exam start will be considered plagiarism
- No external help! You cannot ask questions online or in discussion groups
- No discussion of the exam (or sharing of code) with anyone except for COMP1511 subject staff
- Effectively, you can use whatever is on the internet at 1pm at the start of the exam, but nothing that is added afterwards
- We will have an email address to contact us during the exam: cs1511.exam@cse.unsw.edu.au

The Exam - Technical Details

You will receive an email at your UNSW email address a few days before the exam

Make sure your UNSW email address is working!

This email will contain a link to the exam website
The link will start working at 1pm (13:00 sydney time) on the day of the exam
The commands available in the practice exam will all be available during the final exam
Possibly with different names: 1511 fetch-pracexam will be 1511 fetch-exam
The Exam Format

- The following details might change, but only slightly
- 20 short answer “theory” questions (1 mark each)
  - Some multiple choice and others a very small amount of text
- 8 practical questions (10 marks each)
  - Practical questions will involve actual programming
  - Very similar to Lab/Revision questions

Exams - tips

How to survive a (take-home) exam

- Drink water. Dehydration lowers brain functions
- Eat decent meals. Blood sugar also affects your brain, especially in a stressful situation
- Take breaks! Humans can’t think well for more than an hour at a time.
- You don’t need to work for all 6 hours. The exam was designed for 3 hours of consistent work
- You will reach a point where you can’t score any extra marks by spending more time. It’s ok to hit this point and finish
- If you want to prepare, you can look at the list of topics in COMP1511 and write down your own notes in advance for how you want to approach certain types of questions
- Draw Diagrams for any questions you need to think about
- I’d say “chill out, this isn’t a big deal” but no one will believe me

Short Answer Questions

Quick Questions (1 mark per question)

- These questions will be about whether you understand core coding concepts and the C programming language
- Your answers will either be multiple choice or short answers
- Some are: “What will this code do?”
- Some are: “How does this concept work?”
- Some examples are in the Week 10 Lab

Short Answer Questions - How To

- You will fetch a file called exam_mc.txt to answer them in
- This file is in a special format
- Type your answers within the {{{ triple curly brackets }}}
- Only answers in the {{{ triple curly brackets }}} will be accepted
- Some questions will have validation (so, you might only be able to answer with the letters ‘A’, ‘B’, ‘C’, ‘D’ for example)

It will have the same structure as the file prac_mc.txt in the week 10 lab.
Short Answer Questions - tips

These Questions are a “warmup”
- Read through them all before answering
- Skim quickly and answer the ones you definitely know
- Then go back to the ones that take some time to think about
- Prioritise! Get the easy marks, then spend time on the ones you’re reasonably sure of
- If you’re unsure, go back to the lecture slides and do a text search for what you think you need
- Hit up lecture videos if you need to relearn something

Practical Questions

Less questions, more work each (10 marks per question)
- Questions are similar to the Revision Exercises and Labs
- Stages of difficulty from basic to extreme challenge
- Some will have provided code as frameworks
- Each question will need to be written, compiled and tested
- You will have access to autotests (but they’re just tests!)
- Harder questions will have less autotests
- There will be no specific style marking, so you don’t need to explain your code in comments

Hurdles

Hurdles must be passed to pass the course
- There’s an array hurdle, question 1 or 3
- There’s a linked list hurdle, question 2 or 4
- You must earn a mark of 50% or more in at least one array hurdle question
- You must also earn a mark of 50% or more in at least one linked list hurdle question
- The simplest thing is to put a serious effort into questions 1 and 2, which will cover both hurdles

Practical Questions - tips

Solving Problems
- Read all the questions before starting
- Pick the easy ones as you read. Most likely the earlier questions
- Prepare! A couple of minutes thinking and drawing a diagram will clarify how you’re going to approach a question
- Use your lab/test practice! Debugging and testing will be important here
- Less questions answered completely is better than more questions partially answered
- Don’t count the number of autotests. Marks are not based on number of tests passed
Questions 1-2 First Hurdles

Fundamental C Programming - similar to Revision Test question 1 or 2

- Create C programs
- Use variables (ints and doubles)
- scanf and printf
- if statements and loops
- Use of arrays of ints/doubles, possibly 2D (q1)
- Use of linked lists of ints/doubles, without insertion or removal (q2)

Example Question 1

Loop through a 2D array and gather some kind of information

Eg: Edit the function even_lines so it loops through all the elements of an array. Print out every line that has exactly one even number.

```
even_lines(int side_length, int numbers[SIZE][SIZE])
```

```
$ ./evens
13 14 15
16 17 18
21 23 25
[Ctrl + D]
13 14 15
```

Example Question 2

Perform some computation on a linked list

Eg: Given a linked list, print the difference between the largest and smallest values.

Edit the function: int biggest_diff(struct node *head)

```
$ ./biggest_diff 5 4 3 2 1
4
$ ./biggest_diff 3 7 3 7 12
9
```

Questions 3-4 Harder Hurdles

More advanced C - similar to a hard Revision Test question 2

- Everything from Questions 1 and 2 as well as . . .
- Looping through possibly more than once
- Testing more difficult conditions and keeping track of more than one concept
- Some insertion/removal with linked lists (q4)
- Working with Arrays (q3)
- Working with Linked Lists (q4)
### Questions 5-6

**Even Harder C - similar to Revision Test question 3**
- Likely using strings (q5)
- Possibly fgets, command line arguments etc
- Manipulate linked lists (adding and removing items etc) (q6)
- Potentially use malloc() and free() with structs and pointers
- Might use an Abstract Data Type
- Again, more complex combinations, and some questions requiring interesting problem solving

### Question 7 and 8

**Challenge Questions for people chasing HDs (10 marks)**
- Everything taught in the course might be in these questions
- Think Challenge Exercises, even some of the hard ones!
- Will also test your ability to break a problem down into its parts
- This week’s lab has a past Question 8 so you can see the difficulty level
- Partial completion of this question will award some marks

### What to study

**A little preparation goes a long way**
- The basics are important!
- A basic knowledge of several topics is better than an extreme level of knowledge in just one
- Know how to use both **arrays** and **linked lists**
- Try some revision questions from the Tutorials or Labs
- The revision exercises on the course webpage are also very useful (this section will be added to the website this week)

### How important are different topics?

**Important**
- Variables, If, Looping, Functions, Arrays, Linked Lists

**Things that you will need to understand the important topics**
- Characters and Strings, Pointers, Structs, Memory Allocation

**Stretch Goals**
- Abstract Data Types and Multi-File Projects
- Recursion
Exam Marking

Most of the marking will be automated

- Make sure your input/output format matches the specification
- Answers for hurdles will also be checked by hand
- Marks will be earnt for correct code, not for passing autotests
- Minor errors, like a typo in an otherwise correct solution, will only result in a small loss of marks
- Results should be ready by around Sep 2.

Special Consideration and Supplementary Exam

- If you attend the exam, it's an indication that you are well enough to sit the exam
- If you are not well enough to sit the exam, apply for Special Consideration and do not attend the exam
- If you become sick during the exam; or you are unable to continue due to circumstances out of your control, let us know via the provided email address (cs1511.exam@cse.unsw.edu.au).
- A supplementary exam will be held between 6-10 September. If you think you will need to sit this exam, make sure you are available.

Online Exam timetable overlap

- A few students have a 24 hour exams (e.g. INFS2621) which overlaps with COMP1511.
  - The exams unit already knows about this overlap
  - As students do not need that full 24 hours for the other exam, this is not considered a timetable clash

If you are in doubt during the exam, ask!

CS1511 Exam timetable overlap

- The exams unit already knows about this overlap
- As students do not need that full 24 hours for the other exam, this is not considered a timetable clash

Email: cs1511.exam@cse.unsw.edu.au