COMP2521 24T1
Course Review and Exam

Kevin Luxa
cs2521@cse.unsw.edu.au

course review
final exam
course evaluation

Course Review

Course Goals

COMP1511

- gets you thinking like a *programmer*
- solve problems by developing programs
- express your ideas in the C language

COMP2521

- gets you thinking like a *computer scientist*
- know a set of fundamental techniques/structures
- able to reason about their applicability/effectiveness

Acquired Knowledge

- data structures: trees, graphs, hash tables, heaps, tries
- data structure/algorithm analysis: time/space complexity
- sorting and searching techniques
- graph algorithms

Acquired Skills

By the end of this course, you should be able to:

- Implement solutions to a wider range of problems
- Analyse performance characteristics of algorithms
- Analyse performance characteristics of data structures
- Make decisions about appropriate data structures and algorithms

Syllabus Detail
(I)

For each specific data type, we considered:

- implementation in C (data structures, functions)
- operations (e.g., insert, search, delete, traverse)
- analysis of efficiency of operations
- applications of the data type

Syllabus Detail
(II)

Recursion

Analysis of algorithms

- empirical analysis
- theoretical analysis
- time complexity and big-O

Syllabus Detail
(III)

Sorting algorithms

- properties
  - time/space complexity
  - stability
  - adaptability
- elementary sorts
  - selection sort
  - bubble sort
  - insertion sort
  - shell sort
- divide and conquer sorts
  - merge sort
  - quick sort
- non-comparison-based sorts
  - radix sort

Syllabus (IV)

ADTs

- interface vs. implementation
- defining ADTs in C
- stacks
- queues
- sets

Trees

- tree terminology
- tree properties
- binary search trees
- balancing binary search trees
- avl trees

Syllabus (V)

Graphs

- graph terminology/properties
- graph representations
- graph traversal
  - bfs/dfs
- graph problems
  - cycle checking
  - connected components
  - hamiltonian/eulerian paths/circuits
- warshall's algorithm
- dijkstra's algorithm
- minimum spanning trees
  - kruskal's algorithm
  - prim's algorithm

Syllabus (VI)

Hash tables

- hash functions
- collision resolution
  - separate chaining
  - linear probing
  - double hashing
- applications

Priority queues and heaps

- implementations
- binary heaps
- heap sort

Tries

- implementations
- applications

Course Assessment

Assessments:

- 15% labs
- 10% quizzes
- 15% assignment 1
- 15% assignment 2
- 45% final exam

To pass COMP2521, you must:

- score at least 50/100 overall
- score at least 18/45 (40%) on the final exam

Final Exam

Final Exam

- 3-hour in-person exam
- Thursday 2nd May
- Invigilated and held in CSE labs
- Closed book - no materials allowed
  - Code/pseudocode for main data structures and algorithms will be available
  - C quick-reference will be available

Final Exam

- Two sessions: morning and afternoon
  - You will be asked to indicate a preference via email
  - Exam organisers will allocate you to your preferred session if possible
  - You will receive an email with your allocation early Week 11
  - Students with a clash will be pre-allocated to appropriate session
  - To prevent communication between students in morning and afternoon sessions:
    - Students in morning session cannot leave early
    - Students in afternoon session will be corralled before the end of the morning session
    - Students in afternoon session are not allowed to be late

Exam Conditions

- UNSW on-campus exam rules apply
  - see `https://www.student.unsw.edu.au/exam/rules`
- Items/materials:
  - You must bring your UNSW student ID card
    - Must not be expired
  - You may bring a clear water bottle
  - You may bring a clear pencil case (or plastic sleeve) with pens/pencils
  - You may not bring your own keyboard/mouse or other hardware
  - All other items must be placed in your bag
  - Phone, smart watch, other electronic devices must be *switched off* and placed in your bag
- Deliberate violation of exam conditions will be treated as serious misconduct and may be referred to the SCIU

Exam Environment

- Restricted environment - **not** your CSE account
- No access to Internet
  - Uh oh, no ChatGPT!
- No access to any of your files
- Available editors: gedit, VSCode, vim
  - All come with syntax highlighting
  - VSCode comes with clangd extension - provides IntelliSense
- Standard CSE lab machine commands available
  - make, clang, gdb, valgrind, man
- Calculator app available
- You get a chance to try out the exam environment during in-person Week 10 labs

Exam Format

- Marked out of 100
- Two sections:
  - Short-answer questions, worth 40 marks
  - Programming questions, worth 60 marks
- Each question answered in a separate file
- Submit answers using the `submit` command
  - Submit as you complete each question
  - Check what you have submitted using the `submit` command

Exam Format

Short-answer questions

- Theory questions
- Tests your knowledge, understanding, critical thinking
  - Proofs not required
- Most questions will require explanation/justification
  - If question does not ask for explanation, then no need
- Questions may have sub-questions
- Each question will specify a file to write your answers in
  - Starter version of file will be provided
  - Each file will clearly indicate where to write answers for each sub-question

Exam Format

Programming questions

- Tests your problem solving and programming ability
- Each question will ask you to implement one function
- Questions will include examples
- Questions will provide sample test cases
  - Passing these test cases means your solution mostly works
- Your solution must attempt to solve the problem generally
  - Solutions that just hardcode return values for provided tests will receive zero
- Each question will specify a file to implement your solution in
  - Starter version of file will be provided
  - Makefile and main program will be provided
  - If solution requires ADT(s), they will be provided

Exam Format

Programming questions

- Helper functions allowed
- Defining your own #defines, structs, enums allowed
- Using any functions provided by #included libraries allowed
- Global/static variables **strictly forbidden**
- Inefficient solutions (within reason) allowed unless specified
- Questions may specify additional constraints
  - E.g., no while loops, for loops, or goto
  - E.g., time complexity must not be worse than $O(n)$
  - Your solution must abide by these constraints or you may receive fewer marks (or zero) for the question

Exam Format

Programming questions

- Solutions will be automarked
- All solutions will be manually inspected
  - To ensure constraints have been followed
- No marks awarded for style
  - But a human marker needs to be able to deduce the behaviour of your program
- Solutions receiving less than 50% from automarking may receive partial marks for making substantial progress towards a correct solution
  - Resulting mark will not be greater than 50%
- Marks awarded for code only - pseudocode or English description is not worth marks

Special Exam Conditions

- Students with extra exam time approved by ELS will be given extra time
  - Handled by CSE Exams team
- Exam paper shows the standard time limit (3 hours), any extra time is additional to it
- Email us if you have any concerns regarding ELS conditions

Exam Clashes

- UNSW policy is that you may be required to take two exams in one day
- Students with clashes will be automatically allocated to a non-clashing session by the CSE Exams team

Special Consideration

- The exam is covered by UNSW's fit-to-sit policy
- By starting the exam:
  - You are saying "I am well enough to finish the exam"
  - You cannot apply for Special Consideration for issues that existed prior to the exam
- If you are unwell before the exam:
  - Do not attend the exam
  - See a doctor and get a medical certificate ASAP
  - Apply for Special Consideration
- If you become unwell during the exam: **talk to an exam supervisor ASAP**

Supplementary Exam

- If you miss the original exam due to illness/misadventure
  - Apply for special consideration - you may be eligible for a supplementary exam
- Supplementary exams will take place between **Monday 20th May** and **Friday 24th May**, and will be in person, just like final exam

Scaling

- Scaling depends on mark distribution
- We manually inspect the work of students just below the pass threshold
  - If we see many students who've sufficiently shown competency with basic course material but have not passed, exam mark may be scaled up

Revision Strategy

How to revise?

- Re-read lecture slides
- Review tutorial questions, lab exercises, quizzes
  - Redo them without looking at answers/solutions
- Do extra lab exercises and practice exercises
- Try to understand/reproduce lecture code
  - Programming is a skill that improves with practice

Course Evaluation

Course Evaluation

Course evaluation via myExperience:

- How did we do?
- What did you like?
- What could be improved?
- Let us know!
  - `https://myexperience.unsw.edu.au`
- Please give your tutors feedback - myExperience is the best way to give them feedback, and it will more likely than not make their day.
- Please give *specific* feedback
  - Specific feedback is more actionable

Course Evaluation

Vague feedback:

- “there could be a lot of things that could have been improved.”
- “some of the topics are difficult to understand”
- “some concepts didn’t really go in-depth and explain the code properly”
- “I would have appreciated better visualisations of certain concepts”

Course Evaluation

Positive Feedback

Positive feedback:

- AVL trees
  - "The demonstration of different cases was really helpful!"
- Graph problems
  - "Good lecture, Kevin is good at explaining stuff"
  - "I liked how Kevin was coding the graph problems live. It helped me understand better"
- Applications of hash tables
  - "Great examples of how to use hash tables to solve problems"
- Priority queues and heaps
  - "Very thorough, good teaching!"
- Lectures in general
  - "These lectures are underrated, it is evident that there is a significant amount of work and effort put into each one." ... "Plus these are much more in-depth and easier to fall back to, imo."
  - "I like the systematic examples on the slides, like when visualising a hash table insertion etc."

Course Evaluation

Possible Improvements

Constructive feedback:

- Analysis of algorithms
  - “Would love to have seen and gone through more examples of code as opposed to the heavy theory behind the algorithms”
- Elementary sorting algorithms
  - “Good lecture but for the shell sort implementation it would be nice if there was an explanation as to why for the outer loop, we would divide h by 3 after each iteration instead of subtracting by 1”
- Digraph algorithms
  - “Honestly took me ages to understand, but I think making a link back to discrete may have helped, with the transitive property - if there is a link from a to b, and b to c, therefore there is a path from a to c.”
- Priority queues and heaps
  - “Could use some more detail on real world applications but thats just me”

Course Evaluation

Incentive for myExperience completion:

- At 50% response rate, and for every 5% after:
  - We will reveal the topic and mark value of a randomly chosen exam question
  - Topic/mark value of one question revealed at 50%, 55%, 60%, 65%, 70%, ...
- Bonus: At 85% response rate, I'll shave my head!

Thank You

A big thank you to:

- Our lovely teaching staff
  - Tutors and lab assistants
  - Forum staff
  - Help session staff
- All of you!
  - For engaging with the course and giving it your all!

Good Luck

Good luck!

We hope what you've learned in this course will be useful.

We hope you get the mark you're aiming for!

And Finally...

Good luck with the exam, and with your future studies!