Cosc 201
Algorithms and Data Structures
Lecture 25 (27/5/2026)
Exam Overview

Brendan McCane
brendan.mccane@otago.ac.nz

UNIVERSITY OF OTAGO
Te Whare Wānanga o Otāgo
NEW ZEALAND

Exam format

- Exam has a total of 75 marks.

Exam format

- Exam has a total of 75 marks.
- Nine questions.

Exam format

- Exam has a total of 75 marks.
- Nine questions.
- Each question has a title, indicating its theme.

Exam format

- Exam has a total of 75 marks.
- Nine questions.
- Each question has a title, indicating its theme.
- Questions are typically divided into three or four parts, and points are specified for each part.

Exam format

- Exam has a total of 75 marks.
- Nine questions.
- Each question has a title, indicating its theme.
- Questions are typically divided into three or four parts, and points are specified for each part.
- In terms of structure and content previous COSC201 exams are a good guide.

Exam format

- Exam has a total of 75 marks.
- Nine questions.
- Each question has a title, indicating its theme.
- Questions are typically divided into three or four parts, and points are specified for each part.
- In terms of structure and content previous COSC201 exams are a good guide.
- Tutorial questions are also a good guide to typical exam questions.

Some general exam-taking advice

- Start each question on a new page (and make the question number clearly visible).

Some general exam-taking advice

- Start each question on a new page (and make the question number clearly visible).
- Be as neat and organized as possible.

Some general exam-taking advice

- Start each question on a new page (and make the question number clearly visible).
- Be as neat and organized as possible.
- Avoid the brain dump technique (almost all parts of questions can be answered fully in a short paragraph at most).

Some general exam-taking advice

- Start each question on a new page (and make the question number clearly visible).
- Be as neat and organized as possible.
- Avoid the brain dump technique (almost all parts of questions can be answered fully in a short paragraph at most).
- Only spend as much time on a question as allocated for marks.

Some general exam-taking advice

- Start each question on a new page (and make the question number clearly visible).
- Be as neat and organized as possible.
- Avoid the brain dump technique (almost all parts of questions can be answered fully in a short paragraph at most).
- Only spend as much time on a question as allocated for marks.
- There are 75 marks and 180 minutes. This is 2 minutes and 24 seconds per mark.

Some general exam-taking advice

- Start each question on a new page (and make the question number clearly visible).
- Be as neat and organized as possible.
- Avoid the brain dump technique (almost all parts of questions can be answered fully in a short paragraph at most).
- Only spend as much time on a question as allocated for marks.
- There are 75 marks and 180 minutes. This is 2 minutes and 24 seconds per mark.
- If you're running out of time on a question, move onto the next question and come back at the end if you have time.

Some general exam-taking advice

- Start each question on a new page (and make the question number clearly visible).
- Be as neat and organized as possible.
- Avoid the brain dump technique (almost all parts of questions can be answered fully in a short paragraph at most).
- Only spend as much time on a question as allocated for marks.
- There are 75 marks and 180 minutes. This is 2 minutes and 24 seconds per mark.
- If you're running out of time on a question, move onto the next question and come back at the end if you have time.
- It's much easier to get partial marks on all questions, than full marks on some of the questions.

What do you need to know?

Everything!

What do you need to know?

Everything!

- In principle, anything covered in lectures, labs or tutorials is fair game.

What do you need to know?

Everything!

- In principle, anything covered in lectures, labs or tutorials is fair game.
- In practice, most of the exam is devoted to material covered in the lectures.

What do you need to know?

Everything!

- In principle, anything covered in lectures, labs or tutorials is fair game.
- In practice, most of the exam is devoted to material covered in the lectures.
- CS in general and programming in particular are cumulative subjects, so a certain amount of background is presupposed (e.g., arrays, references, methods, . . .).

What do you need to know?

Everything!

- In principle, anything covered in lectures, labs or tutorials is fair game.
- In practice, most of the exam is devoted to material covered in the lectures.
- CS in general and programming in particular are cumulative subjects, so a certain amount of background is presupposed (e.g., arrays, references, methods, . . .).
- You will not be asked to write code but you may be asked to write an algorithm or pseudocode.

What do you need to know?

Everything!

- In principle, anything covered in lectures, labs or tutorials is fair game.
- In practice, most of the exam is devoted to material covered in the lectures.
- CS in general and programming in particular are cumulative subjects, so a certain amount of background is presupposed (e.g., arrays, references, methods, . . .).
- You will not be asked to write code but you may be asked to write an algorithm or pseudocode.
- You may be asked to read code and analyse it.

The nine question headings

- Union-find (how to do operations, costs of operations, state of structures after operations)

The nine question headings

- Union-find (how to do operations, costs of operations, state of structures after operations)
- Induction and algorithm analysis (substitution for recurrences, induction on big-O questions)

The nine question headings

- Union-find (how to do operations, costs of operations, state of structures after operations)
- Induction and algorithm analysis (substitution for recurrences, induction on big-O questions)
- Sorting (time complexity for given code with justification)

The nine question headings

- Union-find (how to do operations, costs of operations, state of structures after operations)
- Induction and algorithm analysis (substitution for recurrences, induction on big-O questions)
- Sorting (time complexity for given code with justification)
- Heaps (how to do operations, result of given code/operations)

The nine question headings

- Union-find (how to do operations, costs of operations, state of structures after operations)
- Induction and algorithm analysis (substitution for recurrences, induction on big-O questions)
- Sorting (time complexity for given code with justification)
- Heaps (how to do operations, result of given code/operations)
- Graphs (breadth-first and depth-first, Dijkstra and Prim)

The nine question headings

- Union-find (how to do operations, costs of operations, state of structures after operations)
- Induction and algorithm analysis (substitution for recurrences, induction on big-O questions)
- Sorting (time complexity for given code with justification)
- Heaps (how to do operations, result of given code/operations)
- Graphs (breadth-first and depth-first, Dijkstra and Prim)
- Binary Search Trees (induction proofs on binary tree properties, properties of balanced BSTs, traversals)

The nine question headings

- Union-find (how to do operations, costs of operations, state of structures after operations)
- Induction and algorithm analysis (substitution for recurrences, induction on big-O questions)
- Sorting (time complexity for given code with justification)
- Heaps (how to do operations, result of given code/operations)
- Graphs (breadth-first and depth-first, Dijkstra and Prim)
- Binary Search Trees (induction proofs on binary tree properties, properties of balanced BSTs, traversals)
- Hashing (collisions, chaining, probing, when to use different data structures)

The nine question headings

- Union-find (how to do operations, costs of operations, state of structures after operations)
- Induction and algorithm analysis (substitution for recurrences, induction on big-O questions)
- Sorting (time complexity for given code with justification)
- Heaps (how to do operations, result of given code/operations)
- Graphs (breadth-first and depth-first, Dijkstra and Prim)
- Binary Search Trees (induction proofs on binary tree properties, properties of balanced BSTs, traversals)
- Hashing (collisions, chaining, probing, when to use different data structures)
- Greedy algorithms and dynamic programming (Huffman coding, dynamic programming problems - naive solutions, memoised solutions, DP solutions)

The nine question headings

- Union-find (how to do operations, costs of operations, state of structures after operations)
- Induction and algorithm analysis (substitution for recurrences, induction on big-O questions)
- Sorting (time complexity for given code with justification)
- Heaps (how to do operations, result of given code/operations)
- Graphs (breadth-first and depth-first, Dijkstra and Prim)
- Binary Search Trees (induction proofs on binary tree properties, properties of balanced BSTs, traversals)
- Hashing (collisions, chaining, probing, when to use different data structures)
- Greedy algorithms and dynamic programming (Huffman coding, dynamic programming problems - naive solutions, memoised solutions, DP solutions)
- P and NP

Broad themes of COSC201

- Representations of sets and maps (lists, trees, and hashing)

Broad themes of COSC201

- Representations of sets and maps (lists, trees, and hashing)
- List-based data structures (queues, stacks, and priority queues)

Broad themes of COSC201

- Representations of sets and maps (lists, trees, and hashing)
- List-based data structures (queues, stacks, and priority queues)
- The idea of *traversal* and fundamentals of graphs and trees

Broad themes of COSC201

- Representations of sets and maps (lists, trees, and hashing)
- List-based data structures (queues, stacks, and priority queues)
- The idea of *traversal* and fundamentals of graphs and trees
- The centrality of induction in arguments about program correctness and time complexity.

Broad themes of COSC201

- Representations of sets and maps (lists, trees, and hashing)
- List-based data structures (queues, stacks, and priority queues)
- The idea of *traversal* and fundamentals of graphs and trees
- The centrality of induction in arguments about program correctness and time complexity.
- Recursion

Broad themes of COSC201

- Representations of sets and maps (lists, trees, and hashing)
- List-based data structures (queues, stacks, and priority queues)
- The idea of *traversal* and fundamentals of graphs and trees
- The centrality of induction in arguments about program correctness and time complexity.
- Recursion
  - Things to avoid

Broad themes of COSC201

- Representations of sets and maps (lists, trees, and hashing)
- List-based data structures (queues, stacks, and priority queues)
- The idea of *traversal* and fundamentals of graphs and trees
- The centrality of induction in arguments about program correctness and time complexity.
- Recursion
  - Things to avoid
  - Things to recognise (simple recursion, divide and conquer)

Broad themes of COSC201

- Representations of sets and maps (lists, trees, and hashing)
- List-based data structures (queues, stacks, and priority queues)
- The idea of *traversal* and fundamentals of graphs and trees
- The centrality of induction in arguments about program correctness and time complexity.
- Recursion
  - Things to avoid
  - Things to recognise (simple recursion, divide and conquer)
  - Clever tricks (memoisation)

Broad themes of COSC201

- Representations of sets and maps (lists, trees, and hashing)
- List-based data structures (queues, stacks, and priority queues)
- The idea of *traversal* and fundamentals of graphs and trees
- The centrality of induction in arguments about program correctness and time complexity.
- Recursion
  - Things to avoid
  - Things to recognise (simple recursion, divide and conquer)
  - Clever tricks (memoisation)
  - Avoiding it through dynamic programming when appropriate.

Surprise Task

Please complete the student evaluation survey now.

Quiz question

1. Which of the following are true?
   - A I have completed the student evaluation survey as requested.
   - B I know you asked me to, but TikTok had a bunch of stuff I had to catch up on.
   - C I have philosophical objections to all surveys.
   - D Don't answer D, it's probably wrong.

2. True or False: $1 + 1 = 2$?