哈爾濱工業大學(深圳)
HARBIN INSTITUTE OF TECHNOLOGY, SHENZHEN

数据结构
Data Structures

Final Review

Prof. Yitian Shao
School of Computer Science and Technology

Final Review

What are expected for passing the final test of this course

- Concept of ADT
- ADTs and operation time complexity
- Searching methods and time complexity
- Sorting methods and time complexity

Course Content

Week 1: Introduction (2/25)	Review of C++ Programming (2/27)
Week 2: **Array** and **Matrix** (3/4)	**Basics of Algorithm** (3/6)
Week 3: **Linked List** I (3/11)	**Linked List** II (3/13)
Week 4: **Stack** and **Queue** I (3/18)	**Stack** and **Queue** II (3/20)
Week 5: **String** I (3/25)	**String** II (3/27)
Week 6: **Tree** I (4/1)	**Tree** II (4/3)
Week 7: **Tree** III (4/8)	**Tree** IV (4/10)
Week 8: **Tree** V (4/15)	**Graph** I (4/17)
Week 9: **Graph** II (4/22)	**Graph** III (4/24)
Week 10: **Searching** Algorithm I (4/29)	*(holiday)*
Week 11: **Searching** Algorithm II (5/6)	**Searching** Algorithm III (5/8)
Week 12: **Sorting** Algorithm I (5/13)	**Sorting** Algorithm I (5/15)
Week 13: **Sorting** Algorithm I (5/20)	

Concept and Fundamental C++ Coding Skills

- The concept of Abstract Data Type (ADT)
  - What are atomic data types, what are abstract data types?
- Fundamental C++ programming skills
  - Variable definition and assignment
  - Use of arithmetic, relational, and logical operators
  - Use of functions
  - Use of vector and string
  - Sequence, selection, and iteration code blocks
  - Use of structures and classes

Concept and Practice of Analyzing Time Complexity

- Evaluate an algorithm via analyzing time complexity (Chapter 4 and Chapter 12)
  - Can perform Big O analysis on any given C++ code
  - Complexity classes: constant, logarithmic, linear, log-linear, quadratic
  - The above complexity classes correspond to what ADT modification, ADT searching and sorting algorithms?

Vectors (Chapter 3)

- **vector** is part of the C++ STL; how to use it in C++ programming
  - Element access
  - Check capacity: size, empty
  - Modify a vector: insert, push_back, erase, pop_back
  - Iterate over a vector
  - Create and initialize a vector
  - Create and initialize a vector of vectors (matrix)
  - Access, modify, iterate over matrices.

Linked List (Chapter 5)

- Concept of linked list: storing every element in its own block of memory
- Singly linked list
- Realization of a linked list using C++ struct, the component of a linked list node
- Create and initialized a linked list
- Iterate over a linked list
- Modify a linked list: add to front, add to back, remove node via indexing, remove front node
- Time complexity of modifying a linked list
- The concept of orphaned memory and how to avoid it

Stack (Chapter 6)

- What is LIFO, when to use a stack?
- **stack** is part of the C++ STL; how to use it in C++ programming
  - Create and initialize a stack
  - Check capacity: size, empty
  - Modify a stack: push, pop, top

Queue (Chapter 6)

- What is FIFO, when to use a queue?
- **queue** is part of the C++ STL; how to use it in C++ programming
  - Create and initialize a queue
  - Check capacity: size, empty
  - Modify a stack: push, pop, front

String (Chapter 7)

- C++ **string** is part of the C++ STL; how to use it in C++ programming
  - Create and initialize a string
  - Check capacity: size, empty
  - Modify a string, index a string, concatenate two strings
  - Find a substring pattern inside a string using the find() function
- Understand the basic concept of Knuth-Morris-Pratt algorithm
- Understand the concept of Longest Prefix Suffix (LPS) array
- Can create a LPS array given any string pattern

Tree (Chapter 8 and Chapter 9)

- (Generic) tree structure
  - The definition
  - The components of a tree: root, leaf, parent, child, subtree
  - How to realize a tree ADT using C++ struct
- Forest
  - The definition
  - Can differentiate forest and tree

Binary Search Tree (Chapter 8)

- The steps of performing a binary search over any array/vector
- Can analyze the time complexity of binary search algorithm
- Binary search tree (BST)
  - The properties of a BST
  - How to realize a BST using C++ struct
  - Can construct a BST given the order of elements to be added sequentially to an empty BST
- The concept of tree recursion: the subtree of a tree

Traverse BST (Chapter 8)

- The steps of traversing a BST
  - Traverse and print data of a BST
- Three BST traversal methods: pre-order, in-order, post-order
  - The usage of different traversal methods
  - Can predict the printed results of any traversal method given any BST
- Realize BST functions via (partial or full) traversing and recursion
  - Can realize BST contain() function via C++ code implementation
  - Can realize BST getMin() function via C++ code implementation
  - Can realize BST getMax() function via C++ code implementation
  - Can realize BST add() function via C++ code implementation
  - Can realize BST freeTree() function via C++ code implementation (delete the tree and free the memory allocation)

Tree Balancing (Chapter 9)

- What is a balanced tree? Know the definition and can identify balanced trees
- What is the runtime complexity (Big O) of add()/remove()/contain() of a BST if it is balanced or unbalanced; can analyze the time complexity given any BST structure
- Can rebalance a BST when a newly added node breaks the balance: left rotation, right rotation, left-right rotation, right-left rotation
    - Can realize BST rebalancing (rotation) via C++ code implementation

Heap (Chapter 9)

- What is a complete tree? Know the definition and can identify complete trees
- Heap ADT
  - The definition
  - The properties of min-heap and max-heap
  - The node swapping procedures when enqueue a node: push(), for both min-heap and max-heap
  - The node swapping procedures when dequeue a node: pop(), for both min-heap and max-heap
- **priority_queue** is part of the C++ STL; how to use it in C++ programming
  - Create and initialize a priority queue as either a min-heap or a max-heap (they are created differently)
  - Check capacity: size, empty
  - Modification: push, pop, top

Huffman Coding (Chapter 9)

- Huffman Trees
  - What is Huffman tree
  - Can construct a Huffman tree given any string
- Huffman Coding
  - Understand the concept of Huffman coding; what is the usage?
  - Can obtain Huffman coding based on any Huffman tree
  - Can decode Huffman coding to restore the encoded string, given a Huffman tree

Graph (Chapter 10)

- What is a graph? Know the definition and can identify graphs
- Graph components: nodes, edges; and their meanings
- Can identify the type of graphs: directed/undirected
- Graph representation: adjacency list, edge list, adjacency matrix
  - Can draw the graph based on any graph representation, and vice versa
  - How to realize a graph node using C++ struct
  - How to represent graph nodes using IDs and edges using a matrix

Path and Weights (Chapter 10)

- Weighted Edge
  - Understand the usage
  - How to represent weighted edge using edge list or adjacency matrix
- Path
  - The definition
  - Identify graph properties: reachable, connected (for undirected graph), complete, cyclic
  - Can build a graph with correct properties given a real-world scenario

Path Finding (Chapter 10)

- Depth-first search (DFS)
  - Can illustrate the correct searching steps of a DFS
  - Can solve a DFS problem via C++ code implementation
- Breadth-first search (BFS)
  - Can illustrate the correct searching steps of a BFS
  - Can solve a BFS problem via C++ code implementation, especially for finding the shortest path
  - Understand the concept of least-cost path for weighted graph
  - Understand the principle of Dijkstra's algorithm

Map (Chapter 11)

- Map
  - The concept of map; the components: key, value
  - How to encode data using a map; C++ code implementation of alphabet encoding
- Unordered map
  - **unordered_map** is part of the C++ STL; how to use it in C++ programming
  - Create and initialize an unordered map
  - Check capacity: size, empty
  - Modification: insert, erase, contains
  - Iterate through the unordered map and print the key-value pairs
  - Can improve program (searching) efficiency using unordered map
- Can realize a map containing constant and sequentially-ordered keys using a C++ vector/array

Hash Function (Chapter 11)

- The concept of hash function
- Know how to evaluate the performance of customized hash functions
- Can implement a simple hash function via array/vector (linear) indexing
- Bucket of hash function
  - The concept of bucket and compression map; when it is needed
  - Understand what is a collision; how it occurs
  - How to resolve collision using linear probing or separate chaining
  - How to realize separate chaining using hash node implemented via C++ struct
  - Know how to analyze the time complexity of searching via hash function and separate chaining
  - Understand what is rehashing and how it works; what is load factor

Sorting (Chapter 4 and Chapter 13)

- Know how to draw diagrams to illustrate the sorting procedures, can implement C++ code, can analyze sorting time complexity:
  - Selection Sort
  - Bubble Sort
  - Insertion Sort
  - Merge Sort
  - Quick Sort
  - Heap Sort

IMPORTANT! MUST READ CAREFULLY!

Announcement of Exam Policy

Exam Policy: 8 Strictly Prohibited Actions

Prohibited Actions During Exam:

1. Do **not** use/hide unauthorized materials (e.g., cheat sheets, notes on hands, electronics); all items must be inside your bag and monitored by proctors
2. Do **not** borrow items (e.g., calculators, stationery) from classmates during the exam; instead, ask proctor for help
3. Do **not** look at another student's papers/answers
4. Do **not** communicate with others (verbally or via gestures) during the exam.
5. Do **not** cover your mouth during the exam. If you have to wear a mask, sit in the first row
6. Do **not** leave the exam room without the proctor's permission
7. Do **not** spend more than **4 minutes** on restroom breaks
8. Do **not** take electronics outside the exam room

Any violation of the rules will result in immediate expulsion from the exam and will be deemed as cheating the exam!

Exam Policy: 8 Rules for Pausing/Ending Exam

Exam Pause Protocol (Restroom Break):

9. Raise hand and **must wait for permission** before leaving the exam room
10. Only **one person** may pause and leave the exam room at any given time
11. You must be escorted by a proctor to the restroom
12. Do **not** talk to anyone during the break
13. Do **not** use any electronics during the break

Exam End Protocol

14. If you finish early or when the exam ends, stay seated; do **not** talk, look around, or interact with others until all exam sheets are collected
15. Do **not** take any exam materials outside the exam room, including draft papers
16. Stop writing immediately when the proctor ends the exam

Any violation of the rules will result in immediate expulsion from the exam and will be deemed as cheating the exam!

6 Ways to Uphold Academic Integrity

Take Action to Uphold Academic Integrity:

1. During the exam, only ask the proctors for any assistance (e.g., borrow pens)
2. Keep your eyes only on your own exam sheets
3. Ignore anyone who tries to communicate with you; report to the proctors immediately if someone distracts you (Do not confront them directly)
4. Protect your answers from being copied by others; never leave your exam materials unattended
5. No post-exam discussion until all students have submitted their exam sheets; stop anyone who violates this rule
6. If you suspect cheating, report the misconduct along with the ID/seat location/name to the leading proctor after the exam

Self-Test: Ensure the Exam Policies Is Clear to You

	Scenario	Consequence	Proper Conduct
1	You forgot to bring a pen / eraser / calculator with you but need to use it; you decided to borrow it from other students.	**NO! This is deemed as cheating!**	**Raise your hand** and wait until a proctor borrows it for you.
2	You feel bored during the exam and you chat with others.	**NO! This is deemed as cheating!**	Submit your exam and **leave the exam room**.
3	You finished your exam early. You meet your friend who pause the exam for the restroom (toilet) and chat with them or borrow your electronics.	**NO! This is deemed as cheating!**	**No interaction** is allowed with those who have not submitted the exam.
4	Talk to yourself, sing/make sound during the exam	**NO! This may be deemed as cheating!**	**No mouth movement**
5	The exam time is over (announced by the proctor). You realize that you forget to write down your name/ID and you rush to do it	**NO! This is deemed as cheating!**	**Stop writing immediately** and quietly wait until all exam sheets are collected. Ask proctors to mark your name and ID

Self-Test: Ensure the Exam Policies Is Clear to You

	Scenario	Consequence	Proper Conduct
6	The exam time is over (announced by the proctor). You stand up and put away your pens, you start chatting with your friends while waiting for the exam sheets to be collected	**NO! This is deemed as cheating!**	**Quietly wait** until all exam sheets are collected. Or both of you can leave the exam room and chat.
7	Take your exam sheets or draft papers outside the exam room, even with some blank draft papers	**NO! This is deemed as cheating!**	**Return all** exam sheets and draft papers to the proctor
8	Fill in the exam sheets for others	**NO! This is deemed as cheating!**	**Do not help others** in any situation, the proctors will help!
9	Accidentally/inadvertently look at other people's exam sheets during the test	**NO! This is deemed as cheating!**	**Focus on your own sheets** for the entire exam period

Self-Test: Ensure the Exam Policies Is Clear to You

	Scenario	Proper Conduct
1	You need to go to the restroom during exam	1. Raise your hand and wait for the proctor's approval 2. Don't talk to/interact with anyone during break 3. Come back to the exam in 4 minutes
2	Someone sitting next to you asked you for help during the exam	Do not respond, or both of you will be caught. Ignore them and stay focused on your own sheets. If the person keeps bothering you, raise hand and report to the proctor
3	If you get sick and have to wear a mask	Sit in the front row (first row) assigned by the leading proctor