Intro to Algorithmic Problem Solving – Course Syllabus

Course Title: Intro to Algorithmic Problem Solving

Lectures: Thursdays in June, 6:00 PM – 8:00 PM (June 5, June 12, June 19, June 26)

Exam: Thursday 6:00 PM – 8:00 PM on July 3

Location: Goldberg Computer Science Building, Room 127

Format: In-person

Instructor: Yuhan Fu (yuhan@dal.ca)

🗓️ Course Overview

This course is designed for students with basic programming experience. It introduces fundamental algorithmic techniques and problem-solving paradigms, helping students develop skills in analyzing problems, designing efficient algorithms, and evaluating complexity. Emphasis is placed on practical applications through a combination of lectures and hands-on exercises using an online coding platform.

🧭 Learning Objectives

By the end of this course, participants will be able to:

- Explain common algorithmic paradigms and complexity measures
- Choose appropriate data structures and algorithms for given problems
- Analyze time and space complexity using asymptotic notation
- Implement and debug algorithms on an online judge platform
- Communicate algorithmic solutions clearly and effectively

🗓️ Weekly Breakdown

Week 1: Algorithm Fundamentals and Two Pointers

- **Session Format:** 2-hour instructor-led session
- **Key Concepts:** This week introduces what algorithms are and how to approach problems systematically. You'll learn how to analyze an algorithm's time and space complexity using Big-O notation. We will review essential data structures such as arrays, linked lists, sets, and dictionaries. You will also study the two-pointer technique, including both the left-right pointer method for problems like reverse and partition, and the slow-fast pointer method for cycle detection and linked list manipulation. You will also be introduced to the prefix sum technique, which is a powerful tool for solving range sum queries efficiently.
- **Assessment: Assignment 1**

Week 2: Recursion, Divide and Conquer, Sorting and Binary Search

- **Session Format:** 2-hour instructor-led session
- **Key Concepts:** This week focuses on recursion, including how the call stack operates and how to write recursive functions with well-defined base and recursive cases. In addition, we will analyze common sorting algorithms in detail, including bubble sort, selection sort, and insertion sort, as well as more efficient divide-and-conquer strategies such as merge sort and quicksort. Finally, binary search will be introduced, covering its standard implementation and advanced variations.
- **Assessment:** Assignment 2

Week 3: Math, Bit Manipulation, Greedy, and Dynamic Programming

- **Session Format:** 2-hour instructor-led session
- **Key Concepts:** You will explore foundational math techniques useful in coding problems, such as computing power(n,x), working with modular arithmetic, and identifying number patterns. Bit manipulation will be introduced with an emphasis on XOR logic, and bit shifting. Greedy algorithms will be discussed, along with how to identify whether local decisions lead to global optima.

Dynamic programming will be introduced with a focus on recognizing overlapping subproblems and optimal substructure, and how to build solutions through memoization and tabulation.

- **Assessment:** Assignment 3

Week 4: Tree, Graph, BFS/DFS and Backtracking

- **Session Format:** 2-hour instructor-led session
- **Key Concepts:** This week covers fundamental tree and graph structures, including binary trees and various tree traversal methods such as preorder, inorder, and postorder. You will also learn graph representations including adjacency lists and adjacency matrices. This week also introduces breadth-first search (BFS) for level-order traversal and depth-first search (DFS) in both recursive and iterative forms. Some common algorithms for finding shortest path and Minimum Spanning Tree (MST) will also be introduced. Finally, backtracking will be introduced as a framework for solving exhaustive search problems.
- **Assessment:** Assignment 4

📋 Assessment Breakdown

Component	Weight (%)
Attendance (minimum 3/4)	0% (mandatory)
Weekly Practice Assignments	2.5%*4 = 10%
Soft Skills Assignment	20%
In-person Exam	70%

- **Attendance**
  - Students are required to attend at least **three out of four** in-person sessions to be eligible for course completion.
  - Attendance is **not graded**, but failure to meet this requirement will result in disqualification from earning a certificate.
- **Weekly Practice Assignments (10%)**

- After each lecture, a set of LeetCode problems will be posted on Brightspace.
- Students must complete the assigned problems and submit a screenshot showing successful submission along with their code and time/space analysis.
- Each assignment contributes 2.5% to the final grade.
- Assignments must be submitted by the due date listed below. Late submissions may be penalized according to the late policy outlined in the syllabus.

Assignment	Post Date	Due Date
1	June 5	June 11
2	June 12	June 18
3	June 19	June 25
4	June 26	July 2

- **Soft Skills Assessment (20%)**
  - Students will be required to record and submit a video (maximum 2 minutes) explaining their solution and its time and space complexity to a selected algorithm problem.
  - The problem will be posted on June 27th, and the video submission is due by **July 2nd**.
  - The video will be evaluated based on clarity of explanation, correctness of the algorithm, time and space efficiency, and overall presentation.
- **In-person Final Exam (70%)**
  - Scheduled on July 3rd, 6:00 PM – 8:00 PM.
  - The exam will be closed-book and conducted in person.
  - It will include both algorithm analysis and algorithm design problems:
    - **Algorithm analysis questions** will present a piece of code and ask you to analyze its behavior, identify bugs, or determine its time and space complexity.

- **Algorithm design questions** will require you to design your own algorithm based on a given problem, write the corresponding code, and provide a time and space complexity analysis of your solution.

✅ Completion Criteria

To successfully complete this course, participants must:

- Attend at least **three of the four** sessions
- Submit all assignments
- Participate in the final exam
- Achieve a total score of **70% or higher**

⏰ Late Policy

To receive full credit, all assignments must be submitted by the specified due date.

- **On-time submission**: 100% of the possible score.
- **Late with approval**: If you have a valid reason and contact me by email before the deadline, a one-day extension may be granted.
- **Submission on the following day (with approval)**: 50% of the possible score.
- **More than one day late**: Submissions will **not be accepted** under any circumstances.

You are encouraged to plan ahead and communicate early if unexpected situations arise.