Algorithms (CMPT405/705)

- **Text book and website**

  Algorithm Design by J. Kleinberg and E. Tardos, (Pearson Addison Wesley, 2005).

  Course homepage, https://www.cs.sfu.ca/CourseCentral/705/qgu/

  CourSys link, https://coursys.sfu.ca/2025fa-cmpt-705-x1/

- **Instructor Information**

  Name: Qianping Gu

  Office: TASC I 8029; Email: qgu@cs.sfu.ca

  Office Hours: 10:00-10:50 Mon via zoom (ID:881 1522 3324, Passwd:740768)

- **TA Information**

  Arash Beikmohammadi, Email: aba189@sfu.ca
  Office hours: 8:30-9:30 Tue at ASB 9808, 11:30-12:30 Fri at ASB 9810

  Amanda Tupper, Email: ajt25@sfu.ca
  Office hours: 11-12 Wed and 12-13 Thu at ASB 9810

Grading

- **Assignments 20%; Midterm Tests 30%; Final exam 50%**
- **Students must attain an overall passing grade on the weighted average of exams in the course to obtain a clear pass (C- or better).**
- **Academic honesty**

  Violation of academic honesty may result in a penalty more severe than zero credit for an assignment, test, and/or exam.

Course objects

- **Give a solid theoretical basis for the design and analysis of computing algorithms through introducing algorithmic techniques, algorithm analysis methods and computation models.**
- **By completing this course, students are expected to be able to**
  - **design their own efficient algorithms for commonly computing problems encountered in their research and analyze the efficiency of the algorithms,**
  - **or prove that it is unlikely to have an efficient algorithm for a problem, and find alternative approaches to tackle the problem such as design approximation/randomized algorithms and evaluate these algorithms.**

Tentative Contents

- **Foundations (1 week): Design and analysis of algorithms, computation models, asymptotic order of growth, graphs, formulate problems by graphs.**
- **Algorithm design techniques (2 weeks): greedy, divide-and-conquer, dynamic programming**
- **Network flow (2 weeks): Maximum flow, minimum cut, applications of network flow.**
- **Computational complexity (2 weeks): NP and computational intractability, PSPACE.**
- **Approximation algorithms (2 weeks).**
- **Algorithms for NP-hard problems in special cases (1 week)**
- **Randomized algorithms (2 weeks).**

Prerequisites

- **Basic knowledge on algorithms**
- **Basic knowledge on (discrete) mathematics**
- **Basic knowledge and skills on programming**