Course Outline

Contents

- Course Details
- Course Summary
- Assumed Knowledge
- Student Learning Outcomes
- Teaching Strategies
- Teaching Rationale
- Student Conduct
- Assessment
- Course Schedule
- Resources for Students
- Course Evaluation and Development

Course Details

Course Code	COMP4128
Course Title	Programming Challenges
Convenor	Raveen De Silva (/users/z3372617)
Admin	Raveen De Silva (/users/z3372617)
Classes	**Lectures** : Monday 6-9pm Timetable for all classes (/COMP4128/20T3/timetable)
Consultations	Thursday 11am-12pm (TBC)
Units of Credit	6
Course Website	http://cse.unsw.edu.au/~cs4128/20T3/ (http://cse.unsw.edu.au/~cs4128/20T3/)
Handbook Entry	http://www.handbook.unsw.edu.au/undergraduate/courses/current/COMP4128.html (http://www.handbook.unsw.edu.au/undergraduate/courses/current/COMP4128.html)

Course Summary

This is a course designed to introduce advanced problem solving techniques to those who have already mastered the fundamentals of programming.

A particular emphasis will be made on modifying existing algorithms to solve new problems in a novel way.

Students will not only develop theoretical approaches to challenging problems, but also implement them.

Assumed Knowledge

The pre-requisite for this course is COMP3821 (or COMP3121 and 75 WAM). Familiarity with the algorithms and data structures presented in COMP3821 will aid students in this course, although much of the material is reintroduced.

All code in COMP4128 is in C++. Expert knowledge of C++ from an object-oriented perspective is *not* required. However, students must be able to write, test and debug relatively short programs in C, and familiarity with the C++ Standard Template Library will be required early in the term.

Student Learning Outcomes

Upon completing this course, students will have mastery in applying various algorithms and data structures to solve competition-style problems, including implementation in C++.

Teaching Strategies

- Lectures will introduce the theory behind the algorithms and data structures used in the course, as well as providing examples of applications to problems of varying difficulty.
- Tute/labs will provide students further guided examples, and an opportunity to practice implementing these algorithms and receive guidance on how to solve problems using the concepts introduced in lectures.

Teaching Rationale

This course extends COMP3121/3821 to include applications of problem-solving techniques, in order to code solutions to problems, particularly in the style of programming contests.

As such, the course trains students to solve problems in this field across a variety of topics, including under time constraints.

Student Conduct

The **Student Code of Conduct** (Information (https://student.unsw.edu.au/conduct) , Policy (https://www.gs.unsw.edu.au/policy/documents/studentcodepolicy.pdf)) sets out what the University expects from students as members of the UNSW community. As well as the learning, teaching and research environment, the University aims to provide an environment that enables students to achieve their full potential and to provide an experience consistent with the University's values and guiding principles. A condition of enrolment is that students *inform themselves* of the University's rules and policies affecting them, and conduct themselves accordingly.

In particular, students have the responsibility to observe standards of equity and respect in dealing with every member of the University community. This applies to all activities on UNSW premises and all external activities related to study and research. This includes behaviour in person as well as behaviour on social media, for example Facebook groups set up for the purpose of discussing UNSW courses or course work. Behaviour that is considered in breach of the Student Code Policy as discriminatory, sexually inappropriate, bullying, harassing, invading another's privacy or causing any person to fear for their personal safety is serious misconduct and can lead to severe penalties, including suspension or exclusion from UNSW.

If you have any concerns, you may raise them with your lecturer, or approach the School Ethics Officer (mailto:ethics-officer@cse.unsw.edu.au) , Grievance Officer (mailto:grievance-officer@cse.unsw.edu.au) , or one of the student representatives.

Plagiarism is defined as (https://student.unsw.edu.au/plagiarism) using the words or ideas of others and presenting them as your own. UNSW and CSE treat plagiarism as academic misconduct, which means that it carries penalties as severe as being excluded from further study at UNSW. There are several on-line sources to help you understand what plagiarism is and how it is dealt with at UNSW:

- Plagiarism and Academic Integrity (https://student.unsw.edu.au/plagiarism)
- UNSW Plagiarism Procedure (https://www.gs.unsw.edu.au/policy/documents/plagiarismprocedure.pdf)

Make sure that you read and understand these. Ignorance is not accepted as an excuse for plagiarism. In particular, you are also responsible that your assignment files are not accessible by anyone but you by setting the correct permissions in your CSE directory and code repository, if using. Note also that plagiarism includes paying or asking another person to do a piece of work for you and then submitting it as your own work.

UNSW has an ongoing commitment to fostering a culture of learning informed by academic integrity. All UNSW staff and students have a responsibility to adhere to this principle of academic integrity. Plagiarism undermines academic integrity and is not tolerated at UNSW. Plagiarism at UNSW is defined as using the words or ideas of others and passing them off as your own.

If you haven't done so yet, please take the time to read the full text of

- UNSW's policy regarding academic honesty and plagiarism (https://student.unsw.edu.au/plagiarism)

The pages below describe the policies and procedures in more detail:

- Student Code Policy (https://www.gs.unsw.edu.au/policy/documents/studentcodepolicy.pdf)
- Student Misconduct Procedure (https://www.gs.unsw.edu.au/policy/documents/studentmisconductprocedures.pdf)
- Plagiarism Policy Statement (https://www.gs.unsw.edu.au/policy/documents/plagiarismpolicy.pdf)
- Plagiarism Procedure (https://www.gs.unsw.edu.au/policy/documents/plagiarismprocedure.pdf)

You should also read the following page which describes your rights and responsibilities in the CSE context:

- Essential Advice for CSE Students (https://www.engineering.unsw.edu.au/computer-science-engineering/about-us/organisational-structure/student-services/policies/essential-advice-for-cse-students)

Assessment

Item	Topics	Due	Marks
Problem sets	All topics	Weeks 2-5,7,9-10	48%
Contest 1	Introduction	Week 2	6%
Contest 2	Data Structures, Binary Search, Dynamic Programming	Week 5	6%
Contest 3	Data Structures II, Graphs, Shortest Paths	Week 9	6%
Final Exam	All topics	Exam period	34%

Course Schedule

Week	Lectures	Labs	Problem Sets	Contests	Notes

1	Introduction	Introduction	Set 1 released	-	-
2	Data Structures I	Data Structures I	Set 2 released	Contest 1	-
3	Dynamic Programming	Dynamic Programming	Set 3 released	-	-
4		Binary Search		-	No lecture (Labour Day)
5	Graphs	Graphs	Set 4 released	Contest 2	-
6	Revision Lecture (optional)	-	-	-	Flexibility Week
7	Data Structures II	Data Structures II	Set 5 released	-	-
8	Shortest Paths	Shortest Paths	Set 6 released	-	-
9	Network Flow	Network Flow	Set 7 released	Contest 3	-
10	Mathematics	Mathematics	Set 8 released	-	-

Resources for Students

The course website will contain all relevant resources, and there are no textbooks required for the course.

The following textbooks are suitable for reference:

- Any algorithms textbook, such as Introduction to Algorithms (Cormen, Leiserson, Rivest and Stein)
- Programming Challenges (Skiena and Revilla)
- Competitive Programming 3 (Halim). A free copy of Competitive Programming 1 can be found at CPBook (https://cpbook.net/#CP1details)

Students may find the following online resources helpful:

- CP-Algorithms (https://cp-algorithms.com/) (and the more comprehensive but Russian Emaxx (http://e-maxx.ru/algo))
- Topcoder Tutorials (https://www.topcoder.com/community/competitive-programming/tutorials/)
- There are many helpful blogs, particularly on Codeforces (a partial list here (https://codeforces.com/blog/entry/57282)). Some select choices:
    - Range Trees (https://codeforces.com/blog/entry/15890)

    - Dynamic Programming on Trees (https://codeforces.com/blog/entry/20935)
    - Geometry (https://codeforces.com/blog/entry/48122)
    - Dynamic Programming Optimizations (https://codeforces.com/blog/entry/8219)

Further practice problems can be found in many places, e.g:

- Codeforces (https://www.codeforces.com)
- AtCoder (https://atcoder.jp)
- Topcoder (http://www.topcoder.com/)
- (http://www.topcoder.com/) CodeChef (https://www.codechef.com/)
- USACO (http://train.usaco.org)

Students are encouraged to ask the lecturer or tutors for further resources.

Course Evaluation and Development

This course is evaluated each session using the myExperience system.

In the previous offering of this courses, students noted that the problem sets were time consuming and requested more example problems.

Based on their comments, we have shortened the problem sets and we are trialling a tute/lab format with walkthroughs of example problems.

Resource created 9 months ago (Friday 11 September 2020, 01:06:24 PM), last modified 9 months ago (Monday 28 September 2020, 04:24:56 PM).

Comments

(/COMP4128/20T3/forums/search?forum_choice=resource/53177)

(/COMP4128/20T3/forums/resource/53177)

There are no comments yet.