# CS3391: ADVANCED PROGRAMMING

**Effective Term**
Semester A 2025/26

## Part I Course Overview

**Course Title**
Advanced Programming

**Subject Code**
CS - Computer Science
**Course Number**
3391

**Academic Unit**
Computer Science (CS)

**College/School**
College of Computing (CC)

**Course Duration**
One Semester

**Credit Units**
3

**Level**
B1, B2, B3, B4 - Bachelor's Degree

**Medium of Instruction**
English

**Medium of Assessment**
English

**Prerequisites**
CS2310 Computer Programming or
CS2311 Computer Programming or
CS2313 Computer Programming or equivalent

**Precursors**
Nil

**Equivalent Courses**
Nil

**Exclusive Courses**
Nil

## Part II Course Details

**Abstract**

For many algorithmic problems found in practical systems, the best solutions are usually the elegant combinations of both efficient algorithms and advanced programming techniques. They are the results of some exciting blend of programming, mathematics and problem solving. The course introduces an interesting variety of subjects in programming, algorithms, and discrete mathematics through puzzles and practical problems so that students will have the chance to perform original discovery of new programming challenges and devise new ideas on solving the problems in an innovative way in terms of algorithms and programming techniques. After learning the course, the students will be able to develop advanced algorithmic and programming skills that are required to solve sophisticated problems in the real world. Due to the practicality of the problems which appear in many collegiate programming contests, we expect that the best students from this course will also be competent to solve competition-style programming problems and may be able to represent City University of Hong Kong at ACM Collegiate Programming contests.

**Course Intended Learning Outcomes (CILOs)**

| | CILOs | Weighting (if app.) | DEC-A1 | DEC-A2 | DEC-A3 |
|---|---|---|---|---|---|
| 1 | Analyze programming problems, create new ideas on constructing algorithms and propose programming techniques for solving the problems. | 40 | | | |
| 2 | Write computer programs based on the algorithms devised and programming techniques chosen for solving problems. | 40 | x | x | |
| 3 | Write computer programs to solve problems under time pressure. | 5 | | x | |
| 4 | Generate new approaches on enhancing team programming and problem solving techniques. | 15 | | | x |

A1: Attitude
Develop an attitude of discovery/innovation/creativity, as demonstrated by students possessing a strong sense of curiosity, asking questions actively, challenging assumptions or engaging in inquiry together with teachers.

A2: Ability
Develop the ability/skill needed to discover/innovate/create, as demonstrated by students possessing critical thinking skills to assess ideas, acquiring research skills, synthesizing knowledge across disciplines or applying academic knowledge to real-life problems.

A3: Accomplishments
Demonstrate accomplishment of discovery/innovation/creativity through producing /constructing creative works/new artefacts, effective solutions to real-life problems or new processes.

**Learning and Teaching Activities (LTAs)**

| | LTAs | Brief Description | CILO No. | Hours/week (if applicable) |
|---|---|---|---|---|
| 1 | Lecture | Students will learn algorithms and related examples in lectures. | 1 | 3 hours/week |

| | | | | |
|---|---|---|---|---|
| 2 | Lab | Students can create new ideas and invent new approaches on designing algorithms and computer codes to solve programming problems during the laboratory sessions. In the laboratory sessions, sets of problems are given to students. The students can learn how to analyze the problems and devise strategies with optimal use of resources and time to solve the problems. Based on the algorithms and programming techniques proposed by students or given by instructors, students spend the laboratory sessions to instantiate the algorithms by writing corresponding computer programs.<br>In regular laboratory sessions, students tackle programming problems in teams. Various team formations will be arranged so that students can work with different individuals, so to generate new approaches on enhancing team programming and problem solving techniques. | 2, 3, 4 | 8 hours/semester (2 hours for each session and in total 4 sessions) |
| 3 | Assignments | The students will improve their proficiency on various programming techniques by doing Take-home assignments. | 2 | After Class |
| 4 | Quiz | Students will compete with each other and try to finish as many problems as possible within a limited period of time. | 3 | 2-3 throughout the semester |

**Assessment Tasks / Activities (ATs)**

| | ATs | CILO No. | Weighting (%) | Remarks ("-" for nil entry) | Allow Use of GenAI? |
|---|---|---|---|---|---|
| 1 | Assignments | 1, 2 | 45 | - | No |
| 2 | Quiz | 3, 4 | 15 | May work individually or in teams | No |

**Continuous Assessment (%)**
60

**Examination (%)**
40

**Examination Duration (Hours)**
3

**Minimum Examination Passing Requirement (%)**
30

**Additional Information for ATs**
For a student to pass the course, at least 30% of the maximum mark for the examination must be obtained.

**Assessment Rubrics (AR)**

**Assessment Task**
Assignments

**Criterion**
ABILITY to SOLVE questions using the techniques learned in the lectures

**Excellent (A+, A, A-)**
High

**Good (B+, B, B-)**
Significant

**Fair (C+, C, C-)**
Moderate

**Marginal (D)**
Basic

**Failure (F)**
Not even reaching marginal levels

---

**Assessment Task**
Quiz

**Criterion**
ABILITY to solve problems under time pressure and group collaboration

**Excellent (A+, A, A-)**
High

**Good (B+, B, B-)**
Significant

**Fair (C+, C, C-)**
Moderate

**Marginal (D)**
Basic

**Failure (F)**
Not even reaching marginal levels

---

**Assessment Task**
Examination

**Criterion**
ABILITY to solve problems for different topics under time pressure

**Excellent (A+, A, A-)**
High

**Good (B+, B, B-)**
Significant

**Fair (C+, C, C-)**
Moderate

**Marginal (D)**
Basic

**Failure (F)**
Not even reaching marginal levels

---

# Part III Other Information

**Keyword Syllabus**
Standard libraries in C and C++; C and C++ input and output processing; Recursion; Dynamic programming; Parsing; Graph algorithms; Strings; Search algorithms; Simulation problems; Spanning trees; Sets; Shortest path; Maximum flow; Computational geometry; Arithmetic, Algebra and number theory; Greedy and enumeration algorithms.

**Reading List**

**Compulsory Readings**

| | Title |
|---|---|
| 1 | Steven S. Skiena and Miguel A. Revilla (2003). Programming Challenges: The Programming Contest Training Manual, Springer-Verlag. |

**Additional Readings**

| | Title |
|---|---|
| 1 | Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms. McGraw Hill, first or second edition. |
| 2 | Steven Halim (2013). Competitive Programming 3. Lulu. |