## TQF 3 Course Specifications
## Section 1 General Information

1. Course code and course title

| | | |
|---|---|---|
| Thai | ICCS๒๐๘ | โครงสร้างข้อมูลและการกำหนดรูปแบบลักษณะข้อมูล |
| English | ICCS208 | Data Structures and Abstractions |

2. Number of credits 4 (3-2-7) (Lecture/Lab/Self-study)

3. Program and type of subject

   3.1 Program Bachelor of Science (Computer Science)

   3.2 Type of Subject Required course

4. Course Coordinator and Course Lecturer

   4.1 Course Coordinator Kanat TANGWONGSAN, PhD

   4.2 Course Lecturers Petch SAJJACHOLAPUNT, PhD

5. Trimester/ Year of Study

   5.1 Trimester Once every academic year

   5.2 Course Capacity Approximately 30 students

6. Pre-requisite(s) ICMA101 Introduction to Computer Programming ***or***
ICPY221 Computer Programming for Physics

7. Co-requisite(s) ICCS 206 Discrete Mathematics

8. Venue of Study Mahidol University, Salaya Campus

## Section 2 Goals and Objectives

1. Course Goals

To equip students with the skills and tools to confidently write efficient and correct programs, and to provide them with further practice in problem solving.

2. Objectives of Course Development/Revision

2.1 Course Objectives

This course is designed to fulfill the requirements of TQF1 and the recommendations from the Association for Computing Machinery (ACM).

2.2 Course-level Learning Outcomes: CLOs

By the end of the course, students will be able to (CLOs)

CLO1 Recognize common forms of abstractions in problem solving and object-oriented programming.

CLO2 Design and implement a well-structured Java program using the industry standard toolchain.

CLO3 Use fundamental data structures and algorithms as a basis for implementing an abstract data type and associated algorithms to solve a given problem.

CLO4 Mathematically analyze and empirically evaluate a simple program in terms of space and time.

## Section 3 Course Management

1. Course Description

Principles of object-oriented design and problem solving; Objects and classes; Encapsulation, abstraction, and information hiding; Inheritance and polymorphism; Unit testing; Abstract data types and data structures, including stacks, queues, linked lists, hash tables, ordered dictionaries, binary search trees, priority queues, and heaps; Upper bound efficiency analysis using Big-O; Elements of Java programming; Use of an integrated development environment

หลักการออกแบบเชิงวัตถุและการแก้ปัญหา วัตถุและคลาส การห่อหุ้ม การกำหนดสาระสำคัญ และการซ่อนข้อมูล การรับทอดและภาวะที่มีหลายรูปแบบ การทดสอบหน่วย แบบชนิดข้อมูลนามธรรมและโครงสร้างข้อมูล รวมไปถึง กองซ้อน คิวรายการโยง ตารางแฮช ดิกชั่นนารีแบบเรียงลำดับ ต้นไม้ค้นหาแบบทวิภาค คิวแบบมีสิทธิพิเศษ และฮีป การหาขอบบนประสิทธิภาพโดยใช้บิกโอ องค์ประกอบแห่งภาษาจาวา การใช้สภาพแวดล้อมเพื่อการพัฒนาแบบเบ็ดเสร็จ (ไอดีอี)

2. Credit hours per trimester

| Lecture (Hour(s)) | Laboratory/field trip/internship (Hour(s)) | Self-study (Hour(s)) |
|---|---|---|
| 36 | 24 | 84 |

3. Number of hours that the lecturer provides individual counseling and guidance.

1 hour/week

## Section 4 Development of Students' Learning Outcome

1. Short summary on the knowledge or skills that the course intends to develop in students (CLOs)

By the end of the course, students will be able to:

CLO1 Recognize common forms of abstractions in problem solving and object-oriented programming.

CLO2 Design and implement a well-structured Java program using the industry standard toolchain.

CLO3 Use fundamental data structures and algorithms as a basis for implementing an abstract data type and associated algorithms to solve a given problem.

CLO4 Mathematically analyze and empirically evaluate a simple program in terms of space and time.

2. Teaching methods for developing the knowledge or skills specified in item 1 and evaluation methods of the course learning outcomes

| ICCS227 | Teaching methods | Evaluation Methods |
|---|---|---|
| CLO1 | Reading assignment, interactive lecture, case studies, quiz, group activities, group discussion | Quiz, Homework, Examination |
| CLO2 | Reading assignment, interactive lecture, case studies, quiz, group activities, group discussion | Quiz, Homework, Examination |
| CLO3 | Reading assignment, interactive lecture, case studies, quiz, group activities, group discussion | Quiz, Homework, Examination |
| CLO4 | Reading assignment, interactive lecture, case studies, quiz, group activities, group discussion | Quiz, Homework, Examination |

## Section 5 Teaching and Evaluation Plans

1. Teaching plan

| Week | Topic | Number of Hours | | Teaching Activities/ Media | Lecturer |
|---|---|---|---|---|---|
| | | Lecture Hours | Lab/Field Trip/Internship Hours | | |
| 1 | ● Course Overview, Motivating Examples,<br>● Java Essentials | 3 | 2 | Reading assignment, interactive lecture, quiz, group activities, case studies, group discussion | **Kanat** TANGWONGSAN, PhD<br>**Petch** SAJJACHOLAPUNT, PhD |
| 2 | ● Classes and References<br>● LinkedList: Nested Classes, Sentinel Nodes | 3 | 2 | | |
| 3 | ● ArrayList: Arrays, Resizing, Queue & Stack Discipline<br>● Class Mechanism: Implements, Extends | 3 | 2 | | |
| 4 | ● Higher Order Functions and Subtyping<br>● Build Tool & Some Java Fun | 3 | 2 | | |
| 5 | ● Exceptions, Iterators, and Iterables<br>● Objects and Inductive Thinking | 3 | 2 | | |
| 6 | ● Review and Assessment Week | 3 | 2 | | |
| 7 | ● Performance Characterization: Asymptotics I<br>● Performance Characterization: Asymptotics II | 3 | 2 | | |
| 8 | ● Disjoint Sets<br>● Sorting | 3 | 2 | | |
| 9 | ● Priority Queues and Heaps<br>● Trees and Binary Trees | 3 | 2 | | |
| 10 | ● Maps and Binary Search Trees<br>● Graph Traversal | 3 | 2 | | |
| 11 | ● Minimum Spanning Trees<br>● Randomness in Algorithms | 3 | 2 | | |
| 12 | ● Hashing and Hash Tables<br>● Term Review | 3 | 2 | | |
| | Total | 36 | 24 | | |

## 2. Plan for Assessing Course Learning Outcomes

### 2.1 Assessing and Evaluating Learning Achievement

a. Formative Assessment

- Worksheet
- Class discussion

b. Summative Assessment

(1) Tools and Percentage Weight in Assessment and Evaluation

| **Learning Outcomes** | **Assessment Methods** | **Assessment Ratio (Percentage)** | |
|---|---|---|---|
| CLO1 Recognize common forms of abstractions in problem solving and object-oriented programming. | Homework & Quiz | 5 | 25 |
| | Examination | 20 | |
| CLO2 Design and implement a well-structured Java program using the industry standard toolchain. | Homework & Quiz | 5 | 25 |
| | Examination | 20 | |
| CLO3 Use fundamental data structures and algorithms as a basis for implementing an abstract data type and associated algorithms to solve a given problem. | Homework & Quiz | 5 | 25 |
| | Examination | 20 | |
| CLO4 Mathematically analyze and empirically evaluate a simple program in terms of space and time. | Homework & Quiz | 5 | 25 |
| | Examination | 20 | |
| | | | 100 |

(2) Grading System

| **Grade** | **Achievement** | **Final Score (% Range)** | **GPA** |
|---|---|---|---|
| A | Excellent | 90-100 | 4.0 |
| B+ | Very good | 85-89 | 3.5 |
| B | Good | 80-84 | 3.0 |
| C+ | Fairly good | 75-79 | 2.5 |
| C | Fair | 70-74 | 2.0 |
| D+ | Poor | 65-69 | 1.5 |
| D | Very Poor | 60-64 | 1.0 |
| F | Fail | Less than 60 | 0.0 |

(3) Re-examination (If course lecturer allows to have re-examination)

N/A - (Not applicable with MUIC)

## 3. Student Appeals

N/A

## Section 6 Teaching Materials and Resources

1. Textbooks and/or other documents/materials

- *None; Lecture notes will be provided by the lecturers.*

2. Recommended textbooks and/or other documents/materials

Selected readings from pertinent scientific journals and textbooks or video clips, as posted on the course's e-learning site

3. Other Resources (If any)

N/A

## Section 7 Evaluation and Improvement of Course Management

1. Strategies for evaluating course effectiveness by students

1.1 Student feedback of instructors, teaching methods and materials, and course content through MUIC student evaluation forms

2. Strategies for evaluating teaching methods

2.1 Evaluation of effectiveness based on student evaluation scores and comments

2.2 Evaluation through peer observations by co-instructor or other Division faculty

3. Improvement of teaching methods

3.1 Adjustments based on student feedback, personal observations, comments from peer observations and discussions with supervisor and/or other Division faculty in one-on-one and/or group meetings as specified by MUIC guidelines

4. Verification process for evaluating students' standard achievement outcomes in the course

4.1 Verification through student performance on assessments based on MUIC/Division standards

5. Review and plan for improving the effectiveness of the course

5.1 Course instructors (and coordinator/supervisor) will meet to discuss results of student evaluations and student performance based on learning outcomes in order to identify point for improvement

5.2 Strategy for improvement set according to MUIC/Division guidelines

## Appendix
## Alignment between Courses and General Education courses

Table 1 The relationship between course and Program Learning Outcomes (PLOs)

| | Program Learning Outcomes (PLOs) | | | | | |
|---|---|---|---|---|---|---|
| | PLO1 | PLO2 | PLO3 | PLO4 | PLO5 | PLO6 |
| (ICCS208) | R | | I | | R | I |

Table 2 The relationship between CLOs and Program LOs (Number in table = Sub LOs)

| ICCS208 | Learning Outcomes in the Computer Science Program | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 |
| CLO1 Recognize common forms of abstractions in problem solving and object-oriented programming. | 1.1<br>1.2 | | 3.2 | | | |
| CLO2 Design and implement a well-structured Java program using the industry standard toolchain. | | | | | 5..1 | 6.1 |
| CLO3 Use fundamental data structures and algorithms as a basis for implementing an abstract data type and associated algorithms to solve a given problem. | | | | | 5.1<br>5.4 | |
| CLO4 Mathematically analyze and empirically evaluate a simple program in terms of space and time. | | | | | 5.2 | |

Table 3 The description of Program LOs and Sub LOs of the course

| CS LOs | Sub LOs |
|---|---|
| PLO1 Demonstrate proficiency in scientific communication. | 1.1 Understand the format of communication in computer science. |
| | 1.2 Communicate inchoate ideas to others for further development and refinement. |
| | 1.3 Describe computing concepts to members of the community with accuracy and clarity. |
| PLO2 Carry out work with scientific integrity and professionalism. | 2.1 Recognize the concepts of intellectual property, copyright licenses, and law pertaining to information technology. |
| | 2.2 Provide ethical reasoning and awareness of issues surrounding bias, fabrication, falsification, plagiarism, outside interference, censorship, and information privacy. |
| | 2.3 Demonstrate good time management, self-regulation, autonomy, and professional code of conduct of the discipline. |
| PLO3 Appraise scientific information critically. | 3.1 Apply quantitative reasoning using mathematical methods and scientific facts, taking into consideration multiple perspectives. |
| | 3.2 Provide a succinct description of the issue (i.e., a problem, a question, or a hypothesis), separating facts and assumptions. |
| | 3.3 Differentiate source, validity, objectives, key arguments, and consequences of a piece information. |
| | 3.4 Create a response to the issue by synthesizing collected information critical to the assessment. |
| PLO4 Use a teamwork mindset in the context of computing. | |
| PLO5 Execute common computing methodologies appropriate for a problem scenario. | 5.1 Carry out the process of converting a process/algorithm to a machine-executable program. |
| | 5.2 Use suitable techniques for correctness and cost analysis of computer programs. |
| | 5.3 Deconstruct a computer system to reveal its structure, components, and process of construction. |
| | 5.4 Select common computing techniques (e.g., standard algorithms, data structures, design patterns, programing style, and computing paradigms) appropriate for a given problem scenario. |
| PLO6 Formulate computational solutions to novel situations grounded on the foundation of computer science. | 6.1 Model a given problem using suitable abstractions, including problem decomposition, in the context of computing. |
| | 6.2 Compare the relative strengths and weaknesses among multiple designs or implementations. |

| CS LOs | Sub LOs |
|---|---|
| | 6.3 Assess the feasibility and efficacy of a computational solution based on its design and implementation. |
| | 6.4 Devise computational solutions to novel situations using knowledge and experience in computer science. |