TQF 3 Course Specifications
Section 1 General Information

1. Course code and course title

Thai ICCS๓๑๑ กระบวนทัศน์การเขียนโปรแกรมเชิงฟังก์ชันและเชิงขนาน

English ICCS311 Functional and Parallel Programming

2. Number of credits 4 (4-0-8) (Lecture/Lab/Self-study)

3. Program and type of subject

3.1 Program Bachelor of Science (Computer Science)

3.2 Type of Subject Required

4. Course Coordinator and Course Lecturer

4.1 Course Coordinator Kanat TANGWONGSAN, PhD

4.2 Course Lecturers Rachata AUSAVARUNGNIRUN, PhD

5. Trimester/ Year of Study

5.1 Trimester Once every academic year

5.2 Course Capacity Approximately 30 students

6. Pre-requisite(s) ICCS208 Data Structures and Abstractions

7. Co-requisite(s) -

8. Venue of Study Mahidol University, Salaya Campus

Section 2 Goals and Objectives

1. Course Goals

To provide students with a basic understanding of functional and parallel programming techniques, as well as their applicability to a wide range of real-world applications.

2. Objectives of Course Development/Revision

2.1 Course Objectives

This course is designed to fulfill the requirements of TQF1 and the recommendations of 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	Develop specifications, implement well-typed functional programs, and prove them correct using rigorous techniques.
CLO2	Use proper abstractions, functional idioms, and constructs to structure code with clear and well-designed interfaces.
CLO3	Identify and exploit opportunities for parallelism in code by selecting appropriate decomposition techniques, parallel primitives, algorithms, and function designs.
CLO4	Analyze sequential and parallel programs using cost semantics and parallel cost models such as work and span.

Section 3 Course Management

1. Course Description

Functional evaluation and operational semantics; Recursive functions, cost analysis, and proofs by induction; Datatypes, pattern-matching, and structural recursion/induction; Higher-order functions and currying; Laziness and streams; Cost semantics and parallel cost models such as work and span; Theoretical efficiency and basic scheduling; Parallelism, including parallel decomposition, tree parallelism, and vector parallelism; Shared-memory parallel programming, including OpenMP, fork/join parallelism; Standard parallel primitives and algorithms, including prefix scan, map, reduce, and sorting

การประเมินผลการทำงานและวากยสัมพันธ์การปฏิบัติงาน ฟังก์ชันย้อนกลับ การวิเคราะห์ต้นทุน และ การพิสูจน์โดยอุปนัย ประเภทของข้อมูล การจับคู่รูปแบบและการย้อนกลับและอุปนัยเชิงโครงสร้าง ฟังก์ชันคำสั่งระดับสูงและเคอร์รี่อิ้ง เลซิเนสและสตรีม วากยสัมพันธ์ต้นทุนและต้นแบบต้นทุนคู่ขนาน ได้แก่ งาน และ ส่วนต่อขยายประสิทธิภาพเชิงทฤษฎีและการจัดตารางขั้นพื้นฐาน ทฤษฎีคู่ขนาน การแยกคู่ขนาน ทฤษฎีคู่ขนานแผนภูมิต้นไม้ทฤษฎีคู่ขนานแบบเวคเตอร์ การโปรแกรมมิ่งคู่ขนานแบบหน่วยความจำใช้ร่วมกัน, รวมทั้งโอเพ่นเอ็มพี, ทฤษฎีคู่ขนานแบบร่วม ปฐมฐานเชิงคู่ขนานแบบมาตรฐานและอัลกอริทึม รวมทั้งการค้นหาปัจจัย แผนที่ การลดลง และการค้นหา

2. Credit hours per trimester

Lecture (Hour(s))	Laboratory/field trip/internship (Hour(s))	Self-study (Hour(s))
48	0	96

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 Develop specifications, implement well-typed functional programs, and prove them correct using rigorous techniques.

CLO2 Use proper abstractions, functional idioms, and constructs to structure code with clear and well-designed interfaces.

CLO3 Identify and exploit opportunities for parallelism in code by selecting appropriate decomposition techniques, parallel primitives, algorithms, and function designs.

CLO4 Analyze sequential and parallel programs using cost semantics and parallel cost models such as work and span.

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

ICCS311	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	Functional evaluation and operational semantics	4	0	Reading assignment, interactive lecture, quiz, group activities, case studies, group discussion	TBA
2	Recursive functions, cost analysis, and proofs by induction	4	0		
3	Datatypes, pattern-matching, and structural recursion/induction	4	0		
4	Higher-order functions and currying	4	0		
5	Laziness and streams	4	0		
6	Cost semantics and parallel cost models such as work and span	4	0		
7	Theoretical efficiency and basic scheduling	4	0		
8	Parallelism, including parallel decomposition, tree parallelism, and vector parallelism	4	0		
9-10	Shared-memory parallel programming, including OpenMP, fork/join parallelism	8	0		
11-12	Standard parallel primitives and algorithms, including prefix scan, map, reduce, and sorting	8	0		
	Total	48	-		

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 Develop specifications, implement well-typed functional programs, and prove them correct using rigorous techniques.	Homework & Quiz	5	25
	Examination	20	
CLO2 Use proper abstractions, functional idioms, and constructs to structure code with clear and well-designed interfaces.	Homework & Quiz	5	25
	Examination	20	
CLO3 Identify and exploit opportunities for parallelism in code by selecting appropriate decomposition techniques, parallel primitives, algorithms, and function designs.	Homework & Quiz	5	25
	Examination	20	
CLO4 Analyze sequential and parallel programs using cost semantics and parallel cost models such as work and span.	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
(ICCS311)			R		R	I

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

ICCS311	Learning Outcomes in the Computer Science Program					
	1	2	3	4	5	6
CLO1 Develop specifications, implement well-typed functional programs, and prove them correct using rigorous techniques.			3.2		5.1 5.2	
CLO2 Use proper abstractions, functional idioms, and constructs to structure code with clear and well-designed interfaces.					5.4	6.1
CLO3 Identify and exploit opportunities for parallelism in code by selecting appropriate decomposition techniques, parallel primitives, algorithms, and function designs.					5.4	6.3
CLO4 Analyze sequential and parallel programs using cost semantics and parallel cost models such as work and span.					5.2	6.3

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.