2309 LP002 Course Outline

Subject Code	:	LP002
Subject Title	:	Data Structures
Course Type	:	Compulsory
Level	:	2
Credits	:	4
Teaching Activity	:	Lecture 24 hours
	:	programming 24 hours
Prior Knowledge*	:	CN103: Computer Programming I (variables, functions, arrays, …) CN104: Computer Programming II (pointers, memory, file I/O)

Class	Week	Time	Classroom	Date
D1	Mon	15:30-18:20	C309	2022/09/01-2022/12/14

Instructor	:	Shenlu Jiang
E-mail Address	:	shenlujiang@must.edu.mo
Office	:	A324
Office Hour	:	Mon 15:30-19:00 Tue 12:30-15:00 Wed 14:30-17:30 Thu 12:30-15:30

COURSE DESCRIPTION

This course introduces some basic data structures and algorithm analysis skills. It aims at teaching students the method of organizing data and estimating the running time of algorithms when they develop programs. Topics covered include representation and implementations of abstract data types, elementary algorithm analysis skills, list, stacks, heap, queues, hash-table, binary trees, and graphs, recursive techniques, and sorting algorithms.

TEXT BOOK

Required Text Book:
Book title: Data Structures and Algorithm Analysis in C
Author/Editor: Mark Allen Weiss
Edition: 2
ISBN-10: 0201498405
ISBN-13: 978-0201498400
Publisher: Addision-Wesley Professional
Date: 1997

Reference Book:
1. Book title: Data Structures: A Pseudocode Approach with C
Author/Editor: Richard F. Gilberg, Behrouz A. Forouzan
Edition: 2
ISBN-10: 0534390803
ISBN-13: 978-0534390808
Publisher: Course Technology
Date: 2005

INTENDED LEARNING OUTCOMES

Upon the successful completion of this course students will be able to:

1. Establish a C/C++ project to implement solutions.
2. Estimate the time complexity of algorithms.
3. Write simple recursive/ iterative algorithm.
4. Define and implement simple abstract data type in the .h and .c files.
5. Apply classic abstract data types (e.g., list, stack, and queue) in programming problems.
6. Explain the ideas of classic sorting algorithm.
7. Differentiate stack, queue, heap, and priority queue.
8. Select the proper sorting algorithm for a problem.
9. Write both static and dynamic implementations of lists, stacks, queues, and heap.

Weekly Schedule

Index	Topic	Hours	Teaching Method
1	Introduction	3	Lecture
2	Algorithm analysis	3	Lecture
3	Linear list (concept, ADT, implementation)	3	Lecture/lab
4	Linked list (singly/doubly/circularly –linked list)	3	Lecture/lab
5	Linked list example	3	Lecture/lab
6	Stack & queue	3	Lecture/lab
7	Recursion	3	Lecture/lab
8	Tree	3	Lecture/lab
9	AVL tree	3	Lecture/lab
10	Heap & PriQueue	3	Lecture/lab

11	Hash Table	3	Lecture/lab
12	Sorting 1 (comparison-based sorting)	3	Lecture/lab
13	Sorting 2 (non-comparison-based sorting)	3	Lecture/lab
14	Graph: topological sort	3	Lecture/lab
15	Graph: Traversal	3	Lecture/lab
16	Exam review	3	Lecture

ASSESSMENT APPROACH

Assessment method	**Percentage(%)**
1. Attendance	10%
2. Project and Assignments	60%
3. Final Exam	30%
Total	100%

Guideline for Letter Grade:

Marks	Grade
93 - 100	A+
88 - 92	A
83 - 87	A-
78 - 82	B+
72 - 77	B
68 - 71	B-
63 - 67	C+
59 - 62	C
56 - 58	C-
53 - 55	D+
50 - 52	D
Below 50	F

Notes:
Students will be assessed on the basis of continuous assessment (i.e. coursework in the form of individual assignments and group work) and by an end of semester examination.

The coursework assessment items (e.g. assignment, project, and presentation etc.) evaluate students' ability to implement programming solution to practical problems using applying the data structures and algorithms.

Final examination will primarily evaluate students' overall understanding of main topics covered in the course (i.e., how to organize data and analyse algorithms when they develop programs).

ADDITIONAL READINGS

Journals:
Conferences: Sigcomm, Mobicom, Infocom
Trade and other Publications:
Website:
All source codes used in teaching and guidelines are available on the course ftp site.