Aror University of Art, Architecture, Design & Heritage Sukkur.

BS(Artificial Intelligence)

Fall-2025

Data Structures (Theory + Lab)

Course Title:	Data Structures (Theory + Lab)
Course Code:	CSC-221
Credit Hours:	(3+1)
Course Instructor:	Abdul Khalique
Electronic mail:	akhalique.faculty@aror.edu.pk

Course Objectives

The objective of this course is to make students familiar with the concepts of the way data is stored inside computer and its manipulation using different algorithms. Students will learn different data structures such as array, stack, queue, linked list, trees, graphs, sorting algorithm etc. Since Programming fundamentals is the pre-requisite of this course, therefore, in class we would be using java language to implement all the data structures. However students may use any programming language.

Assessment:

S. No	**Assessment Activities**	Percentage	Total Activities
1.	Sessional: Quizzes/ Assignments (Quizzes, Assignments, & Test)	30%	4
2.	Mid Term Exam	30%	1
3.	Final Exam	40%	1

Course content:

Week No	Topics	Chapters
1,2	• Introduction to the course • What is data structure? o Need of data structures • Elementary data structures • Arrays o Review of single-dimension arrays o Concept and implementation of 2D arrays ▪ Manipulating matrices using arrays o Basic concepts of Multi-dimensional arrays o What are limitations of Arrays?	See the chapter 3.1 in Michael T. Goodrich, Data Structures & Algorithms in Java
3	• Linked lists o Arrays vs. Linked list • Types of linked list o Singly linked list o Circular singly linked list o Doubly linked list o Circular doubly linked list • Defining the Node class • Linked Lists Functions Printing linked list in reverse order using recursion	See the chapter 3.2, 3.3, 3.4 in Michael T. Goodrich, Data Structures & Algorithms in Java See the chapter 10.2 in [CLRS] Thomas H. Cormen, Introduction to Algorithms
4	• Applying dictionary operations on linked lists o Traversing a linked list o Inserting new node ▪ at the head ▪ at any location o Searching a node o Removing a node ▪ from the head ▪ from anywhere • Clearing a linked list	Handouts
5	• Introduction to Queues • The Queue data structure • Application of queues • Array Representation of Queue o Algorithm for Addition of an Element to the Queue o Algorithm for Deletion of an Element to the Queue • Dynamic Representation of Queues Using Linked Lists	See the chapter 6.2 in Michael T. Goodrich, Data Structures & Algorithms in Java See the chapter 10.1 in [CLRS] Thomas H. Cormen, Introduction to Algorithms

	• Circular Queue-Array Representation	
6	• The FIFO structure • Queue operations • Extended queue operations • Dictionary operations on queues • The priority queues • The LIFO structure • Introduction to the stack data structure • Applications of stack • Stack operations • Stack specifications ○ List and arrays ○ Stacks ○ Reversing a list • Stack implementation ○ Using arrays ○ Using linked list • Methods of stack ○ Push ○ Pop • Push down stack	See the chapter 6.1 in Michael T. Goodrich, Data Structures & Algorithms in Java See the chapter 10.1 in [CLRS] Thomas H. Cormen, Introduction to Algorithms
7	• What is algorithm? • Complexity of algorithm ○ Time complexity ○ Space complexity • Analysis of algorithms • Big O Notation ○ Best-case analysis ○ Worst-case analysis • Average-case analysis • Recursion	Handouts
8-9	• Trees Introduction • Tree terminology • Tree Traversal • Concept of Binary Trees • Why use binary trees • Basic Operations • Complete Binary Tree • Priority Queues: Heaps • Max-Heap	See the chapter 8 in Michael T. Goodrich, Data Structures & Algorithms in Java See the chapter 10.4 and 6 in [CLRS] Thomas H. Cormen, Introduction to Algorithms
10-11	• Concept of Binary Search trees and how they work • Finding a node in a binary search tree • Inserting a node • Recursively traversing the tree in In order, Pre	See the chapter 12 in [CLRS] Thomas H. Cormen, Introduction to Algorithms

	and Post order • Applications of tree traversing in sorting	See the chapter 11.1 in Michael T. Goodrich, Data Structures & Algorithms in Java
12-13	• Deleting a node in a Binary Tree with all three cases • Efficiency of Binary Trees • Handling duplicate nodes in BST • Applications of BST • Coding a complete message • Balanced and unbalanced trees • The AVL trees Overview	See the chapter 11.3 in Michael T. Goodrich, Data Structures & Algorithms in Java
14-15	• Simple sorting o Understanding why sorting is important o Bubble sort o Selection sort o Insertion sort o Merge Sort o Quicksort o Efficiency of Quicksort	See the chapter 7 in [CLRS] Thomas H. Cormen, Introduction to Algorithms See the chapter 12 in Michael T. Goodrich, Data Structures & Algorithms in Java
16	• Hashing • Applications of Hashing • Direct Address • Chain based Scheme • Hash Tables	See the chapter 11 in [CLRS] Thomas H. Cormen, Introduction to Algorithms See the chapter 10.2 in Michael T. Goodrich, Data Structures & Algorithms in Java
	• Graphs o Introduction o Searches (DFS & BFS)	See the chapter in Michael T. Goodrich, Data Structures & Algorithms in Java

Text Book

1. Introduction to Algorithms by Thomas H. Cormen, 3rd edition.
2. Data Structures & Algorithms in Java by Michael T. Goodrich, 6th edition.

Reference Material

1. W3schools Data Structures, using python
2. Think Data Structures: Algorithms and Information Retrieval in Java by Allen B. Downey

Course Learning Outcomes

	Course Learning Outcomes (CLO)

Aror University of Art, Architecture, Design & Heritage Sukkur.

CLO	Description	Domain	BT Level
1	Implement various data Structures and their algorithms and apply them in implementing simple applications.	C	2,3
2	Analyze simple algorithms and determine their complexities	C	4,5
3	Apply the knowledge of data structures to another application’s domain.	C	3
*BT=Bloom’s Taxonomy, C=Cognitive domain, P=Psychomotor domain, A=Affective domain			

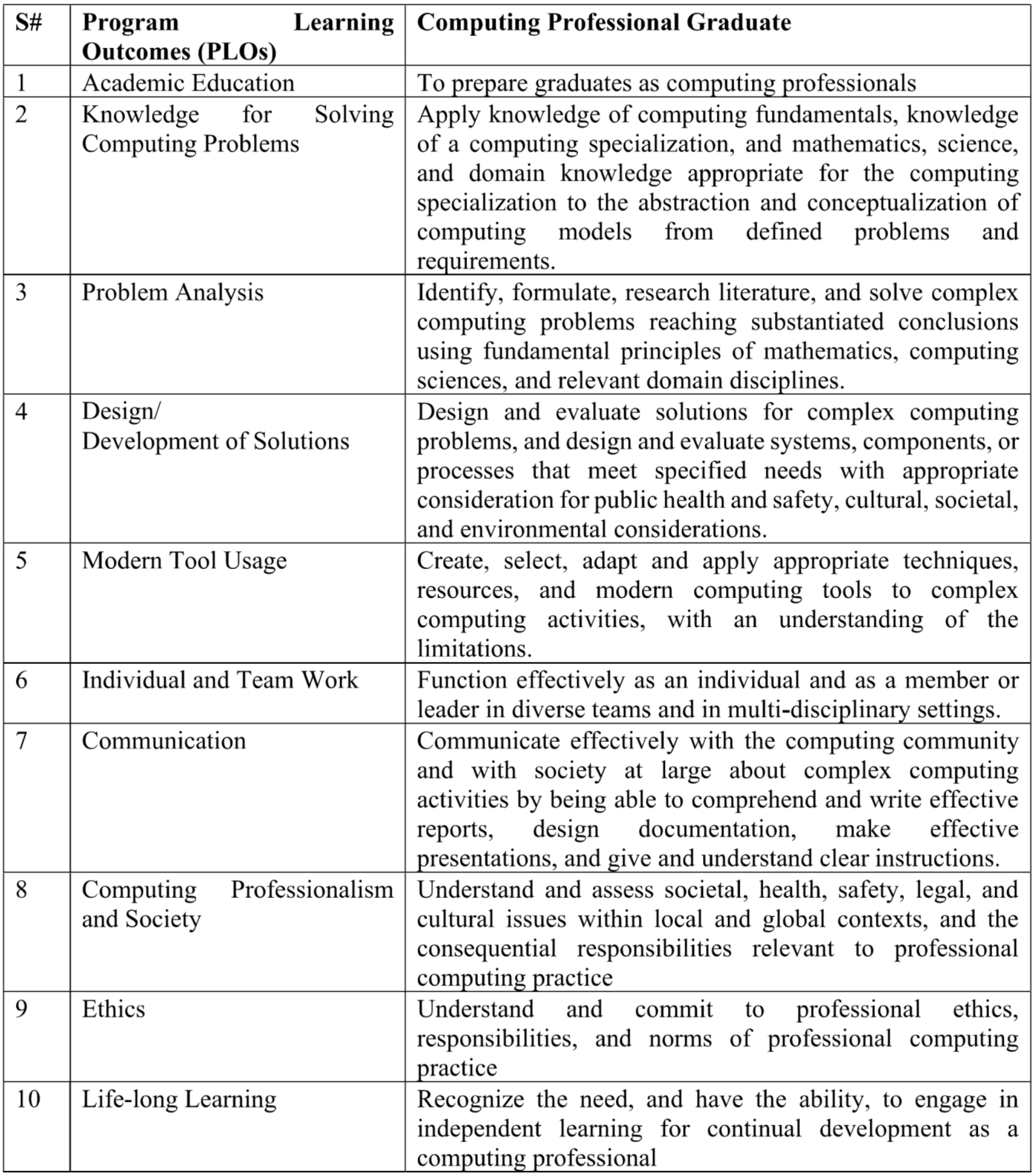
	PROGRAM LEARNING OUTCOMES (PLOs)									
	1	2	3	4	5	6	7	8	9	
CLO.1		X								
CLO.2			X							
CLO.3			X							

Approvals

Prepared By	Mr. Abdul Khalique
Approved By	Not Specified
Last Update	27/8/2025

Program Learning Outcomes

GA: Graduate Attributes



S#	Program Learning Outcomes (PLOs)	Computing Professional Graduate
1	Academic Education	To prepare graduates as computing professionals
2	Knowledge for Solving Computing Problems	Apply knowledge of computing fundamentals, knowledge of a computing specialization, and mathematics, science, and domain knowledge appropriate for the computing specialization to the abstraction and conceptualization of computing models from defined problems and requirements.
3	Problem Analysis	Identify, formulate, research literature, and solve complex computing problems reaching substantiated conclusions using fundamental principles of mathematics, computing sciences, and relevant domain disciplines.
4	Design/ Development of Solutions	Design and evaluate solutions for complex computing problems, and design and evaluate systems, components, or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal, and environmental considerations.
5	Modern Tool Usage	Create, select, adapt and apply appropriate techniques, resources, and modern computing tools to complex computing activities, with an understanding of the limitations.
6	Individual and Team Work	Function effectively as an individual and as a member or leader in diverse teams and in multi-disciplinary settings.
7	Communication	Communicate effectively with the computing community and with society at large about complex computing activities by being able to comprehend and write effective reports, design documentation, make effective presentations, and give and understand clear instructions.
8	Computing Professionalism and Society	Understand and assess societal, health, safety, legal, and cultural issues within local and global contexts, and the consequential responsibilities relevant to professional computing practice
9	Ethics	Understand and commit to professional ethics, responsibilities, and norms of professional computing practice
10	Life-long Learning	Recognize the need, and have the ability, to engage in independent learning for continual development as a computing professional