COURSE DESCRIPTION FORM: CS218: Data Structures

INSTITUTION FAST School of Computing, National University of Computer and Emerging Sciences, Islamabad Campus

PROGRAM(s) TO BE EVALUATED BS-CS: **Fall-2021**

Course Description

Course Code	CS-218
Course Title	Data Structures
Credit Hours	3 + 1
Course Instructors	M. Adnan Tariq, Shehreyar Rashid, Rohail Gulbaz
Grading Policy	Absolute Grading
Policy about missed assessment items in the course	Retake of missed assessment items (other than sessional/ final exam) will not be held. Student who misses an assessment item (other than sessional / final exam) is awarded zero marks in that assessment item i.e. late submission will not be accepted. For missed sessional/ final exam, exam retake/ pretake application along with necessary evidence are required to be submitted to the department secretary. The examination assessment and retake committee decides the exam retake/ pretake cases.
Course Plagiarism Policy	Plagiarism in project or sessional/ final exam will result in F grade in the course. Plagiarism in an assignment will result in zero marks in the whole assignments category.
Prerequisites by Course(s) or Topics	Computer Programming / Object Oriented Programming
Assessment Instruments with Weights (homework, quizzes, sessional exams, final exam, assignments, etc.)	Assessment with the weight.

Assessment Type	**Weight**
Assignments (3)	15
Quizzes (>4)	05
Midterm Exam	15
Lab Work	15
Project	10
Final Exam	40

Course Coordinator	M. Adnan Tariq
URL (if any)	
Course Catalog Description	Data Structures are core to any "Computer Science" degree and are related to efficient storage of information in memory. In this course, we will cover elementary data structures and associated algorithms to manipulate them.

	The main objective of the course is to teach the students how to select and design data structures and algorithms that are appropriate for problems that they might encounter. This course is also about showing the correctness of algorithms and studying their computational complexities. Topics include lists, stacks, queues, arrays, trees, graphs, sorting and searching. This course offers the students a mixture of theoretical knowledge and practical experience. This includes sequential storage (lists, queues, and stacks), hierarchical storage (trees), and association/adjacency storage (graphs). Students will also become familiar with algorithm analysis and design techniques.
Textbook(s)	D.S. Malik, Data Structures Using C++, 8th Edition Mark Allen Weiss, Data Structures and Algorithm Analysis in C++, 4th Edition James Robergé, Stefan Brandle, David Whittington, A laboratory course in C++ Data structures, 2nd Edition
Reference Material	Mark Allen Weiss, Data Structures and Problem Solving Using C++, 2nd Edition Y. Langsam, M. J. Augenstein, A. M. Tenenbaum, Data Structures Using C and C++, 2nd Edition Clifford A. Shaffer, Data Structures and Algorithm Analysis, Edition 3.
Course Goals	**A. Course Learning Outcomes (CLOs)** During this course, students learn strategies and techniques to efficiently store data (Data Structures) and to perform processing on such data in efficient ways (Algorithms), as well as on the analysis and design of such techniques. After successful completion of this course students should be able to: 1. Define basic static and dynamic data structures and relevant standard algorithms for them: Stacks, Queues, Linked lists, Trees, Graphs, Heap, Priority queue, and sorting/searching algorithms 2. Demonstrate advantages and disadvantages of specific algorithms and data structures. 3. Prepare the students to pick the right data structure for a given problem and understand which data structure to implement in a certain scenario. 4. Determine and demonstrate bugs in program, recognize needed basic operations with data structures. 5. Formulate new solutions for programing problems or improve existing code using learned algorithms and data structures. 6. Evaluate algorithms and data structures in terms of time and memory complexity of basic operations. 7. Apply concepts learned in various domains like DBMS, compiler construction etc. **B. Program Learning Outcomes (PLOs)**

1. Computing Knowledge:	Apply knowledge of mathematics, natural sciences, computing fundamentals, and a computing specialization to the solution of complex computing problems.	✔
2. Problem Analysis:	Identify, formulate, research literature, and analyze complex computing problems, reaching substantiated conclusions using first principles of mathematics, natural sciences, and computing sciences.	✔
3. Design/ Develop Solutions:	Design solutions for complex computing problems and design systems, components, and processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal, and environmental considerations.	✔
4. Investigation & Experimentation	Conduct investigation of complex computing problems using research-based knowledge and research-based methods.	
5. Modern Tool Usage:	Create, select, and apply appropriate techniques, resources, and modern computing tools, including prediction and modelling for complex computing problems.	✔
6. Society Responsibility:	Apply reasoning informed by contextual knowledge to assess societal, health, safety, legal, and cultural issues relevant to context of complex computing problems.	
7. Environment and Sustainability:	Understand and evaluate sustainability and impact of professional computing work in the solution of complex computing problems.	
8. Ethics:	Apply ethical principles and commit to professional ethics and responsibilities and norms of computing practice.	
9. Individual and Teamwork:	Function effectively as an individual, and as a member or leader in diverse teams and in multi-disciplinary settings.	
10. Communication:	Communicate effectively on complex computing activities with the computing community and with society at large.	
11. Project Management and Finance:	Demonstrate knowledge and understanding of management principles and economic decision making and apply these to one's own work as a member or a team.	
12. Lifelong Learning:	Recognize the need for and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological changes.	

C. Mapping of CLOs to PLOs	

(CLO: Course Learning Outcome, PLOs: Program Learning Outcomes)

		PLOs											
		1	2	3	4	5	6	7	8	9	10	11	12
CLOs	1	✔	✔	✔									
	2	✔	✔	✔									
	3	✔	✔	✔									
	4	✔	✔	✔									
	5	✔	✔	✔									
	6	✔	✔	✔		✔							
	7	✔				✔							

Topics covered in the course
(assume 15-week instruction and 3 contact hours per week)

1. Topics to be covered:

List of Topics	No. of Weeks	Contact Hours	CLO(s)
Introduction to data structures	**0.5**	**1.5**	**1, 2, 3**
Abstract data types	**0.5**	**1.5**	**1, 2, 3**
Complexity theory	**1**	**3**	**1, 2, 3, 5**
Arrays, Address translation	**0.5**	**1.5**	**1, 3**
Sorting/Searching algorithms	**0.5**	**1.5**	**1, 2, 3**
Lists, Linked list implementation using arrays and pointers	**1.5**	**4.5**	**1, 2, 3**
Queue	**0.5**	**1.5**	**1, 2, 3**
Stacks & applications	**2**	**6.0**	**1, 2, 3, 5**
Trees: Tree Data Structure Definition and Basic terminologies	**0.5**	**1.5**	**1, 2, 3, 5**
Trees: Tree Traversal, Expression trees	**0.5**	**1.5**	**1, 2, 3**
Trees: Binary Search Trees	**1**	**3**	**1, 2, 3**
Trees: AVL Trees	**1.5**	**4.5**	**1, 2, 3**
Graphs: Definitions and Basic terminology	**0.5**	**1.5**	**1, 2, 3**

	Graphs: Traversal	**0.5**	**1.5**	**1, 2, 3**
	Graphs: Minimum spanning trees	**0.5**	**1.5**	**1, 2, 3, 5**
	Graphs: Shortest path trees	**0.5**	**1.5**	**1, 2, 3**
	Hashing	**1.0**	**3.0**	**1, 2, 3**
	Trees: Binary Heap	**0.5**	**1.5**	**1, 2, 3**
	Trees: B-Trees/ B+Trees	**1.0**	**3.0**	**1, 2, 3**
	Total	**15**	**45**	

Programming Language for Assignments (if any)	Implementation of Arrays, Link list, Stack, Queues, Trees and Graph in C++ Language.			
Class Time Spent (in percentage)	**Theory**	**Problem Analysis**	**Solution Design**	**Social and Ethical Issues**
	28	10	5	2
Oral and Written Communications	Every student is required to submit at least __1___ written reports of typically ___5____ pages each and to make __1___ oral presentation of typically ____10___ minutes' duration.			

Lab/ Practical Component of the course

Weeks	Contents/Topics	Assessment Items (Case Study/ Exercise Assignment/ Quiz etc.)
Week-01	ADT, Templates	Lab task 1
Week-02	Templates with Unit testing	Lab task 2
Week-03	2D/3D Arrays with Unit testing	Lab task 3
Week-04	List (Array-based), Recursion with Unit testing	Lab task 4
Week-05	Linked lists (Singly), Recursion	Lab task 5
Week-06	**First Sessional Exam**	
Week-07	Linked list variations (Circular, double), Recursion	Lab task 6
Week-08	Queue	Lab task 7
Week-09	Stacks & Applications	Lab task 8
Week-10	Binary Search Trees (BST)	Lab task 9
Week-11	Binary Search Trees (BST)	Lab task 10
Week-12	**Second Sessional Exam**	
Week-13	AVL Trees	Lab task 11
Week- 14	Graph ADT, Traversals	Lab task 12
Week- 15	Hashing	Lab task 13
Week- 16	Project	